

# Reaping the Benefits of Oracle8i and Oracle9iAS

Just to get (us) started

A correction

LAST September!

A plea

Questions LAST!

An assumption

You know about the technologies:  
HOW rather than WHAT



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Overview

Introduction

Environment

Replication and automated transfers

⊗ XML-chunking and -caching

Function-based indexing

XML to HTML conversion using XSLT

PL/SQL Gateway Cache

interMedia Text

Questions?



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Introduction

### Who? What?

Forlaget Thomson - Karnov & UfR

Primary law with commentaries and case reports.

### Why on-line? Why then (now)? Why Oracle?

E-business or no business.

Stay ahead of the competition ("It's now or never").

Already there! No additional work!?

On-line DB replica of production DB!

Not 24x7!



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Environment

### Sun Solaris with Oracle8i and Oracle9iAS

SunOS 5.8, 8.1.7.2.0, 1.0.2.0 (mod\_plsql 3.0.0.8.3)

### PL/SQL

PL/SQL Web Toolkit and interMedia Text

### XSLT

XML to HTML/CSS

### Java

Single JavaServlet (XDK for Java)



# Reaping the Benefits of Oracle8i and Oracle9iAS

```
PROCEDURE produktfunktioner(vid stam_productversion.versid%TYPE:=NULL)
IS
  reg VARCHAR2(20):='';
BEGIN
  IF NVL(vid,'*')=ilseutil.vidKarnov THEN
    reg:='?reg=Syst.reg.';
  ELSIF NVL(vid,'*')=ilseutil.vidUfr THEN
    reg:='?reg=Kron.reg.';
  END IF;

  ilsehtml.helsidetop('Funktioner');
  http.p('V&aelig;lg mellem f&oslash;lgende funktioner:<br><br>');
  http.prn('<a href="ilseprod.prodframe?frame1url=ilsehtml.topline' ||
    reg || '&frame2url=ilselog.faq" target="thomsondk_main">');
  http.p('<b>FAQ</b></a><br>');
  http.p('Ofte spurgte sp&oslash;rgsm&aring;l og svar!<br><br>');
  ...
  ilsehtml.helsidebund;
END produktfunktioner;
```



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Replication and automated transfers 1/2

### Replication of meta data

#### *Current-awareness!*

Meta data is replicated on an hourly basis.

Single Master to read-only Slave replication.

Deferred and on delete cascade constraints.



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Replication and automated transfers 2/2

### Automated document transfers

#### *Daily new content!*

#### Transformations:

- Unicode/character entities

&#10003; =>

```

```

- (Hyper-)link validation

```
<LR IDREF="LBKG200012.§1">...</LR> =>
```

```
<LR IDREF="LBKG200012.§1" LINKSTAT="INAKTIV">...</LR>
```

- CLOB storage



# Reaping the Benefits of Oracle8i and Oracle9iAS

## XML-chunking and -caching 1/6

No additional manual or large batch- processes

=> Transform the XML to HTML on-the-fly

=> XSLT is the obvious choice

Documents of considerable sizes (many 100Kb's)

=> Serve it in small(er) pieces:

*You may be able to transform it in acceptable time,  
but you cannot send THAT MUCH!*

## Solution

=> XML-chunking (and -caching)

*Algorithm based on existing relational  
database storage structure!*



# Reaping the Benefits of Oracle8i and Oracle9iAS

## *DOCUMENTDOCTYPE*

TOPID -- unique id

## *FAMILY*

TOPID, ID -- (topid, id) is the primary key identifying a node  
MOTHER -- id of parent element (same topid)  
FIRSTCHILD -- id of first child element ("-")  
NEXTSISTER -- id of parents next child ("-")  
BYTESTART -- byte offsets of the first  
BYTEEND and last characters of the node

## *CONTENT*

TOPID, ID, CONTNO -- (topid, id, contno) is the primary key, each  
node in FAMILY has one or more rows here  
CONTENT -- content split at maximum 2000 characters

## *DOCUMENTCONTENT (ONLINE ONLY)*

TOPID

CONTENT -- content as ONE CLOB



# Reaping the Benefits of Oracle8i and Oracle9iAS

## XML-chunking and -caching 3/6

Use FAMILY to calculate where to start and end a chunk. Then go get that piece of XML from DOCUMENTCONTENT.

### Task:

Fast, intelligent algorithm

Well-formed XML

Chunk-navigation

Transformation to HTML (XML to HTML via XSLT)

Highlighting of words (interMedia Text)

### More:

Caching



# Reaping the Benefits of Oracle8i and Oracle9iAS

```
<DOCUMENT NORMID="..."><!-- root element -->
  <CHUNKREFS>
    <CHUNKREF HREF="...?{chunkno-1}" PN="P"/><!--link to prev. chunk-->
    <CHUNKREF HREF="...?{chunkno+1}" PN="N"/><!--link to next chunk-->
  </CHUNKREFS>
  <LOV><!-- appended start-tags to ensure well-formedness -->
    <N1>
      <CHUNKSTART/><!-- HIGHLIGHTING STARTS HERE -->
      <N2><!-- here starts the data from the document -->
        <P>
          ...
        </P><!-- here ends the data from the document -->
      <CHUNKSLUT/><!-- HIGHLIGHTING ENDS HERE -->
    </N2><!-- appended end-tags to ensure well-formedness -->
  </N1>
</LOV>
<CHUNKREFS>...</CHUNKREFS>
</DOCUMENT>
```



1:	<LOV ID="KARL1999281">	(fc=2, m=NULL, ns=NULL, bs=1, be=22)
2:	<L>	(fc=3, m=1, ns=13, bs=23, be=25)
3:	<TITEL>	(fc=4, m=2, ns=12, bs=26, be=32)
4:	Lov	(fc=NULL, m=3, ns=5, bs=33, be=35)
5:	<DATO>	(fc=6, m=3, ns=8, bs=36, be=41)
6:	_1999-05-12_	(fc=NULL, m=5, ns=7, bs=42, be=53)
7:	</DATO>	(fc=NULL, m=5, ns=NULL, bs=54, be=60)
8:	<LOVNR>	(fc=9, m=3, ns=11, bs=61, be=67)
9:	nr._281	(fc=NULL, m=8, ns=10, bs=68, be=74)
10:	</LOVNR>	(fc=NULL, m=8, ns=NULL, bs=75, be=82)
11:	</TITEL>	(fc=NULL, m=3, ns=NULL, bs=83, be=90)
12:	</L>	(fc=NULL, m=2, ns=NULL, bs=91, be=94)
13...:	<NOTER>...</NOTER>	(fc=NULL, m=1, ns=16, bs=95 ... be=112)
16:	</LOV>	(fc=NULL, m=1, ns=NULL, bs=113, be=118)



# Reaping the Benefits of Oracle8i and Oracle9iAS

## XML-chunking and -caching 6/6

### Intelligent algorithm

How do I "well-form"?

=> Know about XML

How do I "stop faster"?

=> Know thy XML - and stop before you go too far

### Transform

Outside database

=> Store and redirect to JavaServlet with storage-id

### Reuse (caching)

They've already been stored!



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Function-based indexing

To create T.O.C. for the documents on-the-fly we used a function-based index, indexing exactly those elements in the XML corresponding to headings.



# Reaping the Benefits of Oracle8i and Oracle9iAS

```
CREATE OR REPLACE FUNCTION pr_niveaunr(sgmlstr VARCHAR2)
  RETURN NUMBER DETERMINISTIC
IS
BEGIN
  IF SUBSTR(sgmlstr,1,3) IN ('<N1','<N2',...,'<N7')
  AND SUBSTR(sgmlstr,4,1) IN ('>',' ') THEN
    RETURN TO_NUMBER(SUBSTR(sgmlstr,3,1));
  ELSE
    RETURN NULL;
  END IF;
END;
/
CREATE INDEX content_inx_niv ON content(topid,pr_niveaunr(content));
ANALYZE TABLE content COMPUTE STATISTICS;
SELECT ... FROM content
  WHERE topid=&&tid
  AND pr_niveaunr(content)=1;
```



# Reaping the Benefits of Oracle8i and Oracle9iAS

## XML to HTML conversion using XSLT

Not PL/SQL!

Not Java in the database (same as PL/SQL)!

JavaServlet:

```
Statement stmt=conn.createStatement();
s="select chunk from documentchunk where chunkid='"+cid+"'";
OracleResultSet rset=(OracleResultSet)stmt.executeQuery(s);
while(rset.next()){
    c_lob=((OracleResultSet)rset).getClob(1);
}
...
DOMParser parser=new DOMParser();
parser.setValidationMode(false);
parser.parse(c_lob.getCharacterStream());
XMLDocument xmlDoc=parser.getDocumentElement();
...

```



# Reaping the Benefits of Oracle8i and Oracle9iAS

## PL/SQL Gateway Cache

A nice little feature introduced with Oracle<sup>9</sup>iAS

`owa_cache`

Enables you to inform Oracle9iAS:

When a page expires.

That a page has not been changed since last time!



# Reaping the Benefits of Oracle8i and Oracle9iAS

```
cacheLevel CONSTANT VARCHAR2(6):='SYSTEM';
PROCEDURE <procedure_name>(<parameter...>) IS
    <variable...>
    etag VARCHAR2(255); elvl VARCHAR2(10);
BEGIN
    <code that needs to be performed every time (logging)>
    etag:=owa_cache.get_etag; elvl:=owa_cache.get_level;
    IF etag IS NOT NULL AND elvl IS NOT NULL THEN -- been here before
        IF etag=TO_CHAR(SYSDATE,'YYYYMMDD') AND elvl=cacheLevel THEN
            owa_cache.set_not_modified; -- today nothing has changed
            RETURN; -- so we won't go get it again
        END IF;
    END IF;
    owa_cache.set_cache(p_etag=>TO_CHAR(SYSDATE,'YYYYMMDD'),
        p_level=>cacheLevel); -- cache new page
    <code that produces new page>
END;
```



# Reaping the Benefits of Oracle8i and Oracle9iAS

## interMedia Text

- 1) Optimal setup  
fields (distinct), prefix
- 2) Optimal utilisation  
no joins, only WITHIN applicable fields  
optimal interMedia query `/*+ FIRST_ROWS */`
- 3) Updating the indexes  
not all the time (ctxsrv)  
once a day (ctx\_ddl.sync\_index)
- 4) Highlighting within chunks
- 5) URL-encoding search strings



# Reaping the Benefits of Oracle8i and Oracle9iAS

## THE CONCLUSION

”Old” technologies:

PL/SQL, replication, automation, relational database lookups using indexes

”New” technologies:

DB: (Temporary) CLOBs, function-based indexes, interMedia Text, AS: mod\_plsql, PL/SQL Gateway Cache, NT: JavaServlets, XML, XSLT

**COMBINE TO REAP!**



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Questions?

What exactly did you mean by...?

Why did you...?

Couldn't you have...?

What on Earth were you thinking about...?

Can I see it live?

Can I have it?

Is it of *any* relevance to me?

But what does it all mean?



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Counts 1/3 (chunking and tranforming)

qaaKARL1999281.xml

201770 - 85850 of NOTER => 115920 bytes

1. chunk: 43473 bytes (§1-§48)

2. chunk: 44098 bytes (§49-§109)

3. chunk: 29585 bytes (§110-§143) => 117156 bytes  
(117156 - 115920 => 1236 bytes (+1%))

qaaKARL1999281.html

1. chunk: 81094 bytes

2. chunk: 88038 bytes

3. chunk: 56908 bytes => 226040 bytes (+93%)



# Reaping the Benefits of Oracle8i and Oracle9iAS

## Counts 2/3 (code)

PL/SQL Units: 27 (me-2)

Lines/bytes of PL/SQL: 18.376/610.618 (87%)

Java Servlets: 1 (DBA)

Lines/bytes of Java (Servlet): 169/5.410 (1%)

XSLT files: 19 (XSLT programmer)

Lines/bytes of XSLT: 1.938/57.833 (9%)

CSS files: 4 (me+XSLT programmer)

Lines/bytes of CSS code: 737/15.304 (3%)



# Reaping the Benefits of Oracle8i and Oracle9iAS

Counts 3/3 (files etc.)

GIF files:

- Icons etc.: 218 - 561.374bytes
- Graphics: 3400 - 118MB

XML Documents: 59.765

Bytes of XML: 1.268.506.683 (avg. 21.225)

Elements: 27.444.742

PDF files: 48.568 - 1.7GB

Users (named): 5.300

