

# You've Got Mail

Leslie M. Tierstein, STR LLC  
Mark Castaldo, STR LLC

# Overview

---

- Architecture, design and code for a robust correspondence generation system
  - Mail merge capability
  - User maintainable
  - Multiple recipients
  - Multiple, user-selectable output formats

## Overview (2)

---

- Functional requirements
- Database design
- Back-end code
- User interface

# Application

---

- Grants Management system for federal agency
- Intranet usage over LAN and WAN for agency employees at HQ and throughout the US
- Internet usage for applicants for grants and current grantees

# Environment

---

- Oracle 8i database (8.1.7)
- Developer 6i - Forms and Reports
- Oracle 9iAS, Forms Server, Reports Server
- Designer 6i
- Windows NT 4
- Internet Explorer 5.5

# Mail Merge Capability

---

- Boilerplate text is defined
- Tokens are embedded
- Tokens represent data to be retrieved from the database
  - Columns
  - Functions applied to columns

# Mail Merge Capability (2)

<<Print Date>>

<<Auth Rep Full Name>>

<<Auth Rep Title>>

<<Legal App Name>>

<<Legal App Addr>>

Dear <<Auth Rep Salut>>:

<<Legal App Name>> is 30 days late in submitting the progress report for grant number <<Grant Nbr>> covering the period ending <<PR Period End Dt>>. This progress report was due on <<PR Due Dt>>.

Failure to provide such reports means that <<Legal App Name>> is out of compliance with the grant requirements. . . .Please submit the progress report immediately.

If you have any questions, please contact your program officer, <<PO Name>> on <<PO Phone>> or by e-mail at <<PO Email>>.



You've Got Mail

# Multiple Letter Types

---

- Printed letter with single recipient/addressee
  - Example: Reminder letter to grantee stating that a progress report is overdue
- Personalized letter to multiple recipients
  - Example: Form letter to each state's governor summarizing the grants awarded in the state
- Internal notification
  - Part of application
  - Simplified inbox



# Setting up Letters

Maintain Letter Text (mnt\_ltrs/cnsuser)

Letter Code  Name  Letter Type   Customizable?  Enclosure?

**Letter Text**

<<Print\_Date>>

<<Auth\_Rep\_Full\_Name>>  
<<Auth\_Rep\_Title>>  
<<Legal\_App\_Name>>  
<<Legal\_App\_Addr>>

Dear <<Auth\_Rep\_Salut>>:

<<Legal\_App\_Name>> is 30 days late in submitting the progress report for grant number <<Grant\_Nbr>> covering the period ending <<PR\_Due\_Dt>> was due on <<PR\_Due\_Dt>>.

Failure to provide such reports means that <<Legal\_App\_Name>> is out of compliance with the grant requirements. The Corporation re on file before amendments or new awards will be made to your organization's grant. Furthermore, the lack of submission may result in

Please submit the progress report immediately.

If you have any questions, please contact your program officer, <<PO\_Name>> on <<PO\_Phone>> or by e-mail at <<PO\_Email>>.

**Notification Text**

PR 30-day late reminder for grant <<grant\_nbr>> has been sent to <<legal\_app\_name>>

Signatory Role

**Available Tokens**

Token Name

- <<A133\_Audit\_Ok\_Cd>>
- <<Action\_Requested\_Cd>>
- <<Afms\_Dir\_Email>>
- <<Afms\_Dir\_Name>>
- <<Afms\_Dir\_Per\_Id>>
- <<Afms\_Dir\_Title>>
- <<Agmt\_Id>>
- <<Agmt\_Purpose\_Cd>>
- <<Agmt\_Type\_Cd>>
- <<Amend\_Nbr>>
- <<Amend\_Txt>>
- <<Amend\_Type\_Cd>>
- <<Applicant\_Type\_Cd>>
- <<Application\_Type\_Cd>>

# Distribution

---

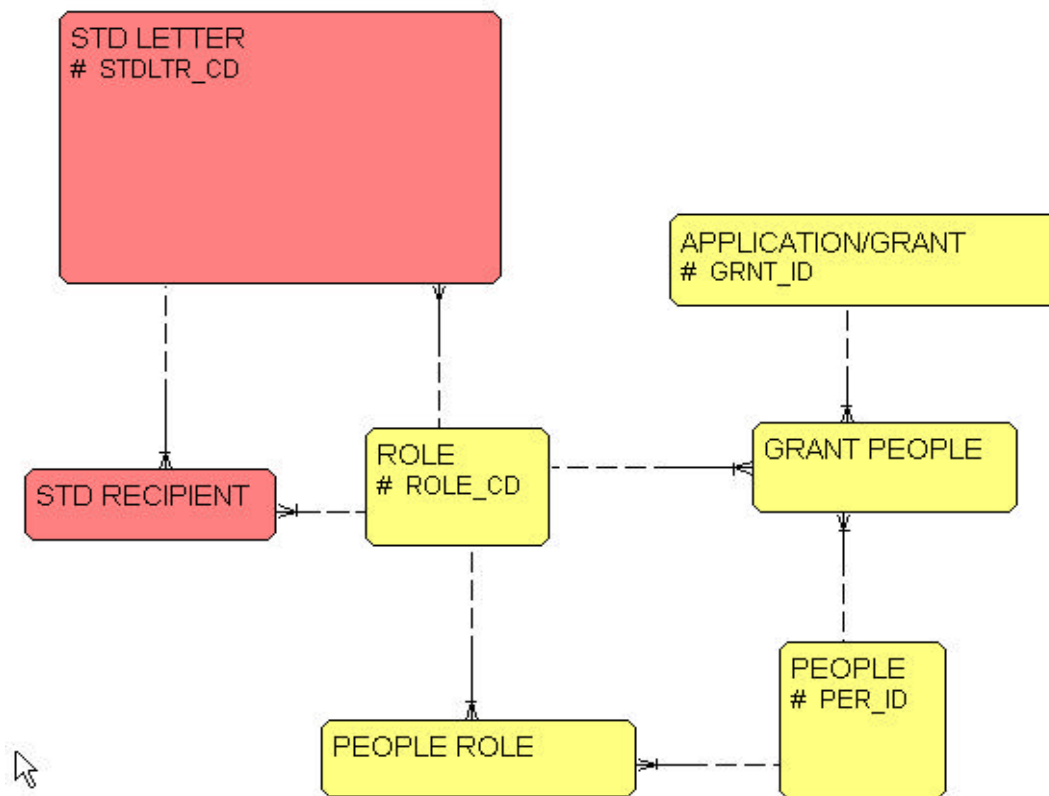
- Multiple recipients/people and roles
  - Recipients are defined by their role in the organization
  - - OR -
  - Their role for a grant
- Software determines the person based on the role
  - Example: “send a notification to the grants officer assigned to this grant”

# Setting up Letter Recipients

- Each recipient has a default/preferred output format
  - Email or printed letter available to all
  - Notification available to internal agency only

Recipient Type	Role		Output Format	
TO	Grantee	↓	Printed Letter	▲
CC	Grants Officer	↓	Notification	
CC	Project Director	↓	Printed Letter	▼

# Database Design – Letter Setup



# Review/Change Output


---


- User may alter text, depending on privileges
  - Letter must be defined as “customizable”
  - User must be the signatory for the letter
- May also alter recipient list and destination type for each recipient
- Can hold letters for a period of time and send later

# Review Output

**Review/Create Correspondence (ent\_corr/cnsuser)**

**Letter Code** PR\_30DL      **Name** 30-Day Late PR Reminder      **Created** 10/27/2001

**Letter Type** Letter w/Single Recipient      **Signator** Ashworth, Robert      

**Application ID** 01ND013657      **Legal Applicant** American Red Cross      **Status** Not Sent      







**Letter Text**


October 27, 2001

Ms. Meridith Drake-Reitan  
Director, Youth and Young Adult Services  
American Red Cross  
2700 Wilshire Blvd  
Los Angeles, CA 90057

Dear Ms. Drake-Reitan:

American Red Cross is 30 days late in submitting the progress report for grant number 00ADNDC001 covering the period ending 12/31/2001. This progress rep 11/01/2001.

Type	Recipient	Output Format
TO 	Drake-Reitan, Meridith	Printed Letter 
CC 	Askew, Keenya	Notification 
CC 	Drake, Meredith	Printed Letter 

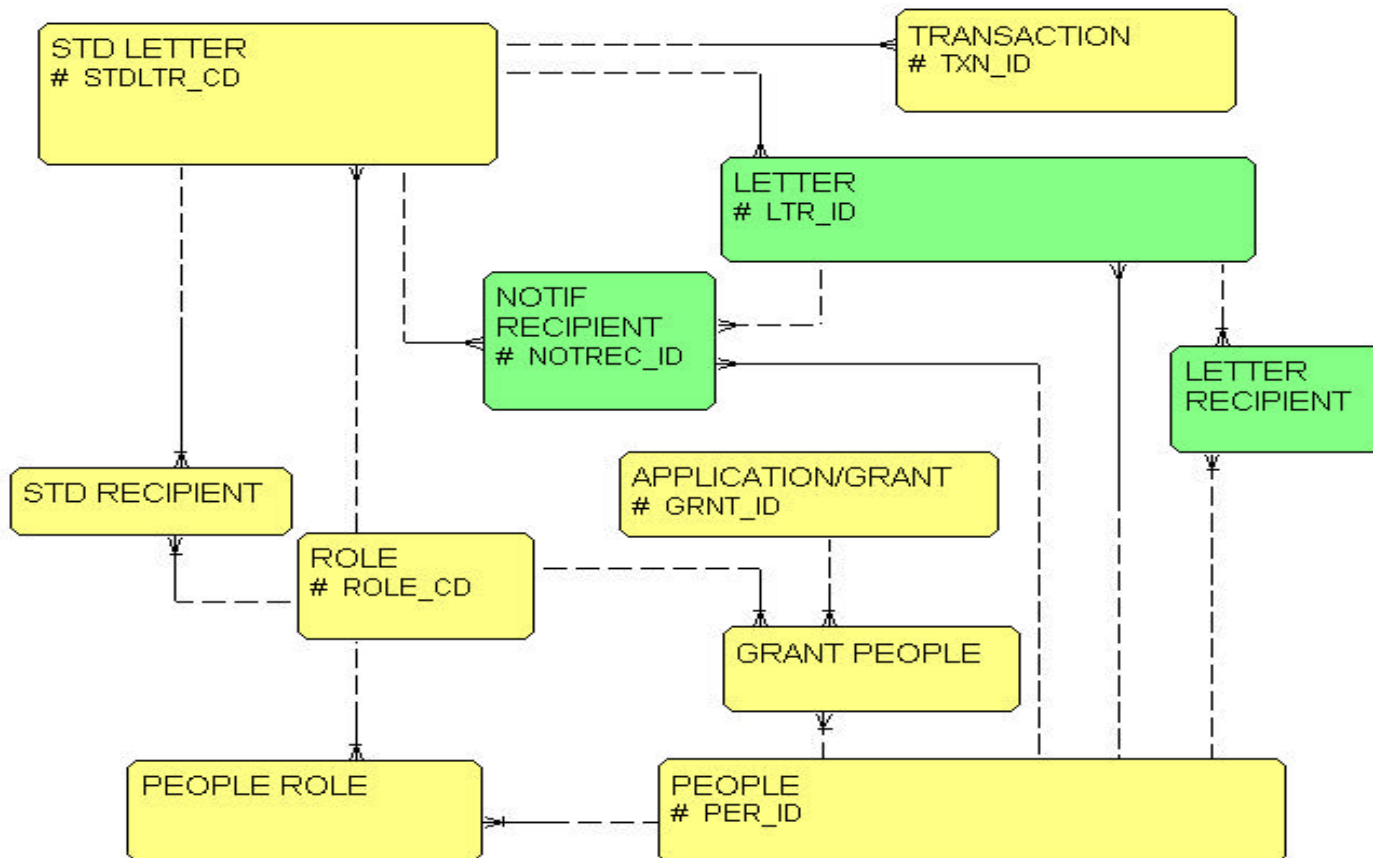


# Letters Generated by:

---

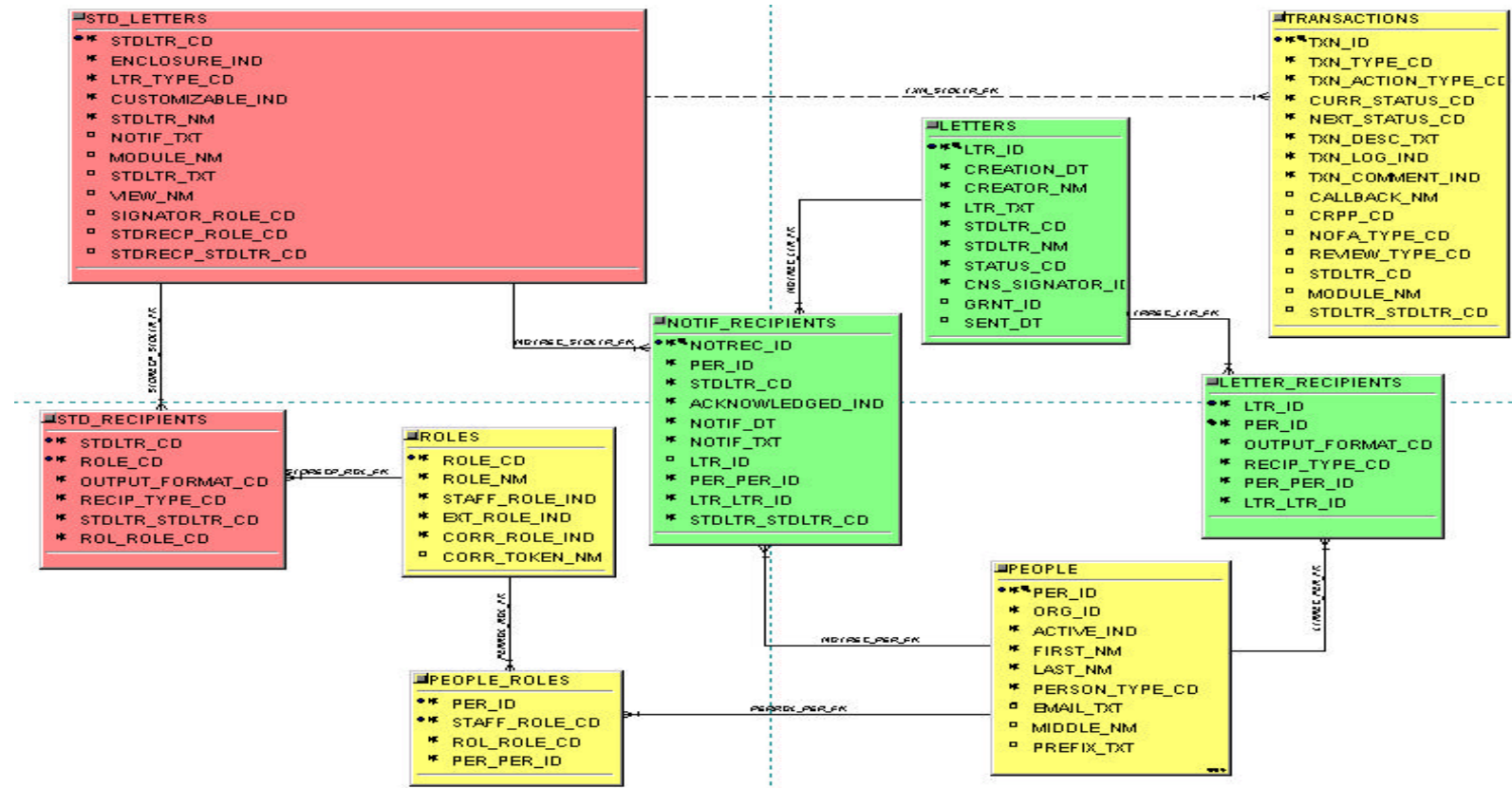
- Event
  - When a grant application is rejected (state transition engine)
- Schedule
  - Send reminder 30 days before progress report is due (DBMS\_JOB)
- Manually
  - User presses “Create Follow-up Letter” button on the Site Visit Results form

# Database Design - Output





# Physical Database Design



# Now what?

---

- Have to actually write code
  - Back-end code to generate letter and recipients
  - Front-end code to allow users to customize letter and recipient list, and send letters
  - Code to deliver the letters
    - Report for printed output
    - Email distribution
    - Notifications for inbox distribution

# Letter Generation

---

- Retrieve information on the current grant and grantee/applicant
  - Use a view to encapsulate information and include calculated columns
- Retrieve the letter text and standard recipients
- Parse the letter text and perform the mail merge

# Database Views

## Source data – Views based on database tables

```
CREATE OR REPLACE VIEW v_corr_pr AS
SELECT
    to_char(SYSDATE, 'fmMonth DD, YYYY')          PRINT_DATE
    , grnt.*
    , grnt.supp_mech_cd                          GRANT_NBR
    , substr(cn_name.format_name
            (arep.prefix_txt, arep.first_nm,
             arep.last_nm, arep.suffix_txt),1,120) AUTH_REP_FULL_NAME
    , arep.title_txt                             AUTH_REP_TITLE
    , to_char (pr.from_dt, 'MM/DD/YYYY')         PERIOD_START_DT
    , pr.pr_status_cd                            PR_STATUS_CD
... FROM cn_grants grnt, cn_people arep,
...WHERE grnt.org_id = legal_app.id
```

# PL/SQL Package

For each view, a cursor and record are defined.

```
PACKAGE cn_corr IS
  CURSOR c_corr (c_grnt_id IN cn_grants.grnt_id%TYPE)
    RETURN v_corr%ROWTYPE;
  r_corr c_corr%ROWTYPE;
  CURSOR c_corr_pr (c_grnt_id IN cn_grants.grnt_id%TYPE,
    RETURN v_corr_pr%ROWTYPE;
  r_corr_pr c_corr_pr%ROWTYPE;

  PROCEDURE gen_ltr (
    p_grnt_id      IN cn_grants.grnt_id%TYPE,
    p_stdltr_cd    IN cn_letters.stdltr_cd%TYPE);
```

# Package Body Components

---

- GEN\_LTR – Driving Procedure
    - Create LETTERS record
    - Create LETTERS\_RECIPIENTS records
      - For each STD\_RECIPIENT...
      - Translate each role into a person
      - Insert a LETTER\_RECIPIENTS record
  - Private procedures
    - GET\_GRANT\_DATA – Retrieve the grant data
    - GET\_LTR\_DATA - Retrieve the letter and recipient data
    - PARSE\_TXT - Parse text
    - XLATE\_TOKENS - Resolve tokens
-

# Retrieve the Source Data

```
PROCEDURE get_grant_data (p_grnt_id    IN cn_grants.grnt_id%TYPE,  
                        p_view_nm    IN cn_std_letters.view_nm%TYPE,  
                        p_rtn_cd     OUT NUMBER,  
                        p_rtn_msg    OUT VARCHAR2) IS  
  
BEGIN  
  IF p_view_nm = 'V_CORR' THEN  
    OPEN c_corr (p_grnt_id);  
    FETCH c_corr INTO r_corr;  
    IF c_corr%NOTFOUND THEN  
      p_rtn_cd := -1;  
    END IF;  
    CLOSE c_corr;  
  ELSIF p_view_nm = 'V_CORR_PR' THEN  
    ...(same)  
  END IF;  
EXCEPTION handle other errors, set return code and message
```

# Retrieve the Letter Data

```
PROCEDURE get_ltr_data ( p_stdltr_cd    IN cn_letters.stdltr_cd%TYPE,  
                        p_letter_txt   OUT VARCHAR2,  
                        p_view_nm      OUT cn_std_letters.view_nm%TYPE,  
                        p_stdltr_nm     OUT cn_std_letters.stdltr_nm%TYPE,  
                        p_signator_token_nm OUT cn_roles.corr_token_nm%TYPE,  
                        p_rtn_cd       OUT NUMBER,  
                        p_rtn_msg      OUT VARCHAR2) IS  
  
  CURSOR c_std_letters (c_stdltr_cd IN cn_std_letters.stdltr_cd%TYPE) IS  
  SELECT dbms_lob.getlength(stdltr_txt) txt_len, stdltr_txt, view_nm,  
         stdltr_nm, rol.corr_token_nm  
  FROM   cn_std_letters stdltr,  
         cn_roles       rol  
 WHERE  stdltr_cd = c_stdltr_cd  
        AND rol.role_cd (+) = stdltr.signator_role_cd;  
  stdltr_rec c_std_letters%ROWTYPE;  
  v_letter_txt  VARCHAR2(32000);
```



## Retrieve the Letter Data (2)

```
BEGIN
  OPEN c_std_letters (p_stdltr_cd);
  FETCH c_std_letters INTO stdltr_rec;
  IF c_std_letters%NOTFOUND THEN
    p_rtn_cd := -1;
  ELSIF stdltr_rec.txt_len > 32000 THEN
    p_rtn_cd := -2;
  ELSE
    p_rtn_cd := 0;
    dbms_lob.read (stdltr_rec.stdltr_txt, stdltr_rec.txt_len, 1, v_letter_txt);
    p_letter_txt := v_letter_txt;
    p_view_nm := stdltr_rec.view_nm;
    p_stdltr_nm := stdltr_rec.stdltr_nm;
    p_signator_token_nm := stdltr_rec.corr_token_nm;
  END IF;
  Close cursor, handle exceptions, etc
```

# Parse Text and Resolve Tokens

---

- Find tokens identified by << >>
- Resolve the token
- Build a single string that now includes boilerplate text and data

# Resolve Tokens

```
PROCEDURE xlate_token ( p_token      IN VARCHAR2,  
                       p_view_nm    IN cn_std_letters.view_nm%TYPE,  
                       p_rtn_txt    OUT VARCHAR2,  
                       p_rtn_cd     OUT NUMBER,  
                       p_rtn_msg    OUT VARCHAR2) IS  
  
  stmt          VARCHAR2(1000);  
  dyn_var       VARCHAR2(4000);  
  v_data_type   VARCHAR2(106);  
  no_data_for_token EXCEPTION;  
  CURSOR c_tab_col ( c_tab_name IN VARCHAR2,  
                   c_col_name IN VARCHAR2) IS  
  
    SELECT data_type  
      FROM user_tab_columns  
     WHERE table_name = c_tab_name  
           AND column_name = c_col_name;
```

## Resolve Tokens (2)

BEGIN

-- Verify that there is a database column that corresponds with the token name.

OPEN c\_tab\_col (p\_view\_nm, p\_token);

FETCH c\_tab\_col INTO v\_data\_type;

*close cursor and handle not found*

IF v\_data\_type <> 'BLOB' THEN

-- build a statement like :dyn\_var := cn\_corr.r\_corr.nofa\_name;

stmt := 'BEGIN :dyn\_var := cn\_corr.r' || ltrim(p\_view\_nm, 'V') ||  
'.' || p\_token || '; end;';

**EXECUTE IMMEDIATE** stmt **USING OUT** dyn\_var;

IF dyn\_var IS NULL THEN

RAISE no\_data\_for\_token;

ELSE

p\_rtn\_txt := dyn\_var;

END IF;

END IF;

# Build Recipient List

```
CURSOR c_recipients (c_stdltr_cd IN cn_std_recipients.stdltr_cd%TYPE) IS
  SELECT output_format_cd, recip_type_cd, corr_token_nm, recip.role_cd
  FROM cn_std_recipients recip,
       cn_roles rol
  WHERE stdltr_cd = c_stdltr_cd
  AND rol.role_cd (+) = recip.role_cd;
```

## Remember...

- Recipients are identified by their role
- Each role has a corresponding token name associated with it

## So...

- Use the same XLATE\_TOKEN routine to translate a role into a person (PER\_ID)
- Insert a row into LETTER\_RECIPIENTS for each one

# Deliver the Mail

- User reviews “outbox”

Query Filter  Not Sent  Sent

Letter Name	Status	Date Sent	Application ID	Grant #	Legal Applicant	Select
30-Day Late PR Reminder	Not Sent		01AS014342	94ASCM0026	MO Community Service Commission	<input checked="" type="checkbox"/>
Application Rejection	Not Sent		01VH000013		City Of Lansing	<input type="checkbox"/>
Application Rejection	Not Sent		01AS014339	94ASCMN024	MN Commission on National and Commur	<input type="checkbox"/>
Application Rejection	Not Sent		01AS014341	94ASCM0026	MO Community Service Commission	<input type="checkbox"/>
30-Day Late PR Reminder	Not Sent		01ND013657	00ADNDC001	American Red Cross	<input type="checkbox"/>
Application Rejection	Not Sent		01VH000015		City Of Lansing	<input type="checkbox"/>
						<input type="checkbox"/>
						<input type="checkbox"/>

## Deliver the Mail (2)

---

- Oracle Reports to produce printed letters
    - LETTERS.LETTER\_TXT is the entire letter body – Easy!
    - Build the signature block by getting the sender's signature image, name, title
    - Build the cc: list based on the LETTER\_RECIPIENTS and translate the PER\_ID into a real name
  - UTIL\_SMTP to send email
  - Internal table for notifications
-

# Send the Mail via UTIL\_SMTP

conn	utl_smtp.connection;	-- smtp connection
smtp_host	VARCHAR2(100);	-- the host name of the smtp server
port	PLS_INTEGER;	-- the port that the smtp server runs on
cr	VARCHAR2(2) := chr(13)  chr(10);	-- carriage return/line feed
reply_rec	utl_smtp.reply	-- a single response from a smtp command
replies_tab	utl_smtp.replies;	-- table of responses from a smtp command

```
smtp_host := 'mail.strllc.com'; -- Don't hard code this
port := to_number('25'); -- standard mail port
```

```
reply_rec := utl_smtp.open_connection(smtp_host, port, conn);
reply_rec := utl_smtp.helo(conn, smtp_host);
```



## Send the Mail via UTIL SMTP (2)

```
FUNCTION send (p_sender_email IN cn_people.email_txt% TYPE,  
              p_recip_email_tab IN recip_email_tab_type,  
              p_subject      IN cn_letters.stdltr_nm% TYPE,  
              p_ltr_txt      IN VARCHAR2) RETURN VARCHAR2;  
  
    v_msg  VARCHAR2(32767);  
BEGIN  
    v_msg := 'Date: ' || TO_CHAR(SYSDATE, 'dd-Mon-yy hh24:mi:ss') || cr ||  
            'From:' || p_sender_email || cr ||  
            'Subject:' || p_subject || cr;  
  
    reply_rec := utl_smtp.mail(conn, p_sender_email);
```

# Send the Mail via UTIL\_SMTP (3)

```
FOR i IN 1..p_recip_email_tab.count LOOP
    reply_rec := utl_smtp.rcpt(conn, p_recip_email_tab(i).email_txt);
    -- Append TO: email@someplace.com
    -- or   CC: email@someplace.com
    v_msg := v_msg || initcap(p_recip_email_tab(i).recip_type_cd) || ':' ||
             p_recip_email_tab(i).email_txt || cr;
END LOOP;
v_msg := v_msg || p_ltr_txt;

reply_rec := utl_smtp.data(conn, v_msg);

utl_smtp.quit(conn);
```

# Send Notifications instead of Mail

**Notification Status Counter [vw\_notif/cnsuser]**

Query Filter  Acknowledged  Not Acknowledged  Both

Notification Type	Count
Concept Paper Accepted	5
New Application Submitted to Headquarters	5
New Application Submitted to State Office	2

Date	Application ID	Grant #	Legal Applicant	Program Name	Ack	
10/16/2001	01VH000015		City Of Lansing	CITY OF LANSING	<input type="checkbox"/>	Go
10/16/2001	01VH000016		City Of Lansing	New one	<input type="checkbox"/>	Go
10/27/2001	01VH000015		City Of Lansing	CITY OF LANSING	<input type="checkbox"/>	Go
					<input type="checkbox"/>	Go

**Notification** The concept paper submitted by City Of Lansing has been accepted for NOFA AmeriCorps\*VISTA - State (2001)

# Limitations/Issues

---

- 32k letter text – PL/SQL VARCHAR2 limitation
- Text only – no pretty formatting
- Each view must be defined in the package.  
Therefore, developer action is required if new views are required.

# Conclusion

---

- Mail generation can be built using custom-written PL/SQL and standard Oracle built-in packages.
- 100% user-maintainability can be achieved – *almost*
- Parser and token resolution code is reusable for other applications, eg, code generation
- Native Dynamic SQL is the key

# About the Authors

---

- Leslie Tierstein is a Technical Project Manager at STR LLC in Fairfax VA. She can be reached at [ltierstein@strllc.com](mailto:ltierstein@strllc.com).
- Mark Castaldo is a Senior Developer at STR LLC in Fairfax VA. He can be reached at [mcastaldo@strllc.com](mailto:mcastaldo@strllc.com)
- This presentation is available on line at: <http://home.earthlink.net/~ltierstein> and at <http://www.strllc.com>.