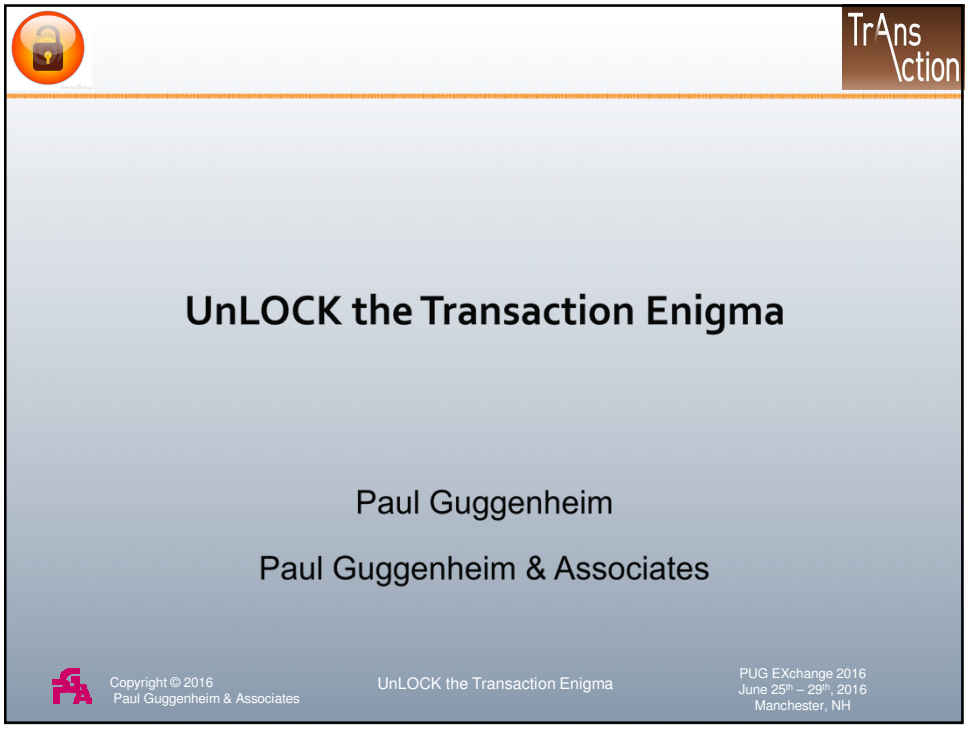
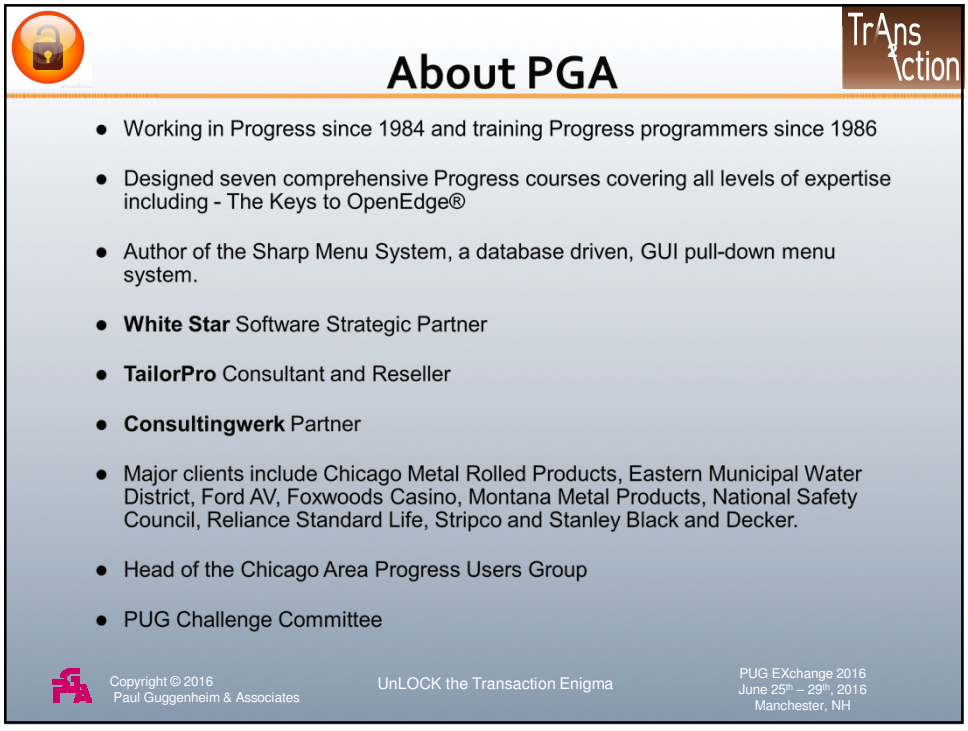

TrAns
ction
UnLOCK the Transaction Enigma
Paul Guggenheim
Paul Guggenheim & Associates
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

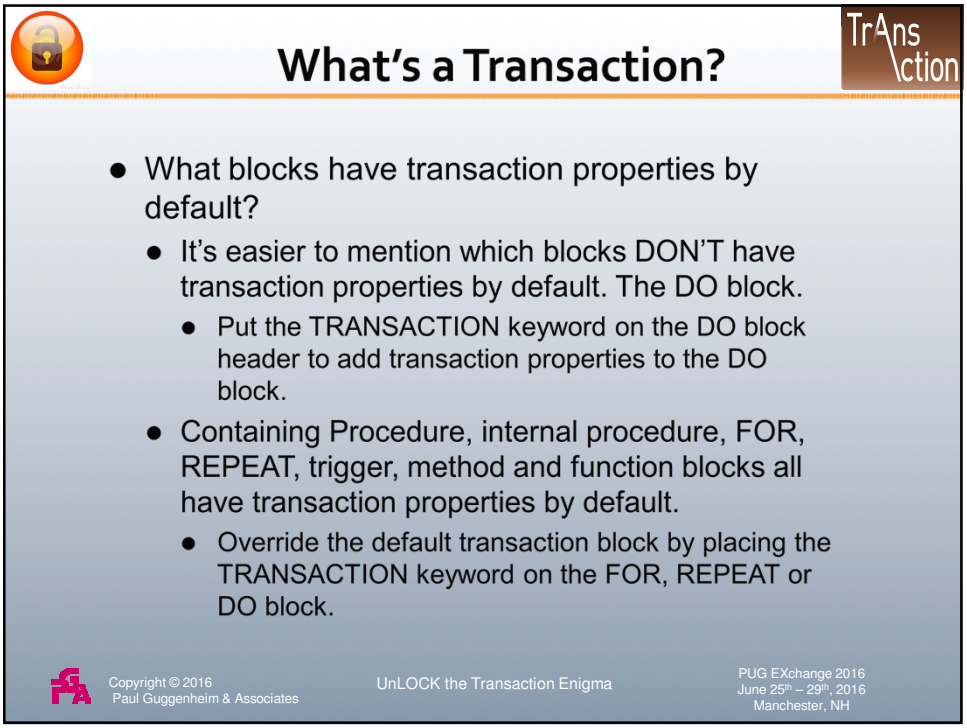

TrAns
ction
About PGA
Working in Progress since 1984 and training Progress programmers since 1986
Designed seven comprehensive Progress courses covering all levels of expertise including - The Keys to OpenEdge®
Author of the Sharp Menu System, a database driven, GUI pull-down menu system.
White Star Software Strategic Partner
TailorPro Consultant and Reseller
Consultingwerk Partner
Major clients include Chicago Metal Rolled Products, Eastern Municipal Water District, Ford AV, Foxwoods Casino, Montana Metal Products, National Safety Council, Reliance Standard Life, Stripco and Stanley Black and Decker.
Head of the Chicago Area Progress Users Group
PUG Challenge Committee
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

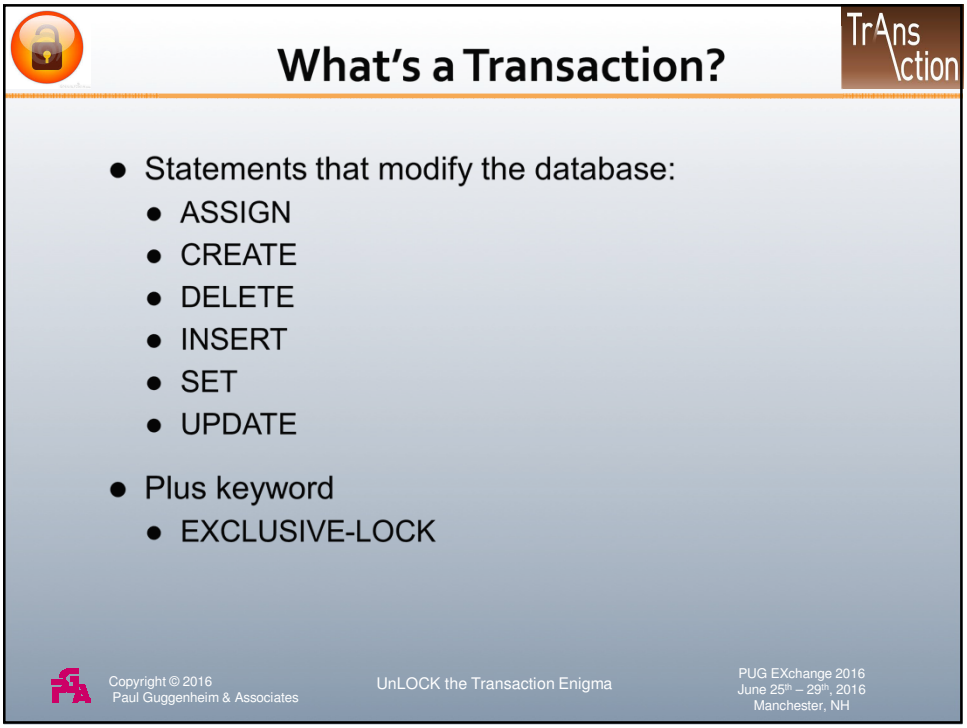
Roadmap

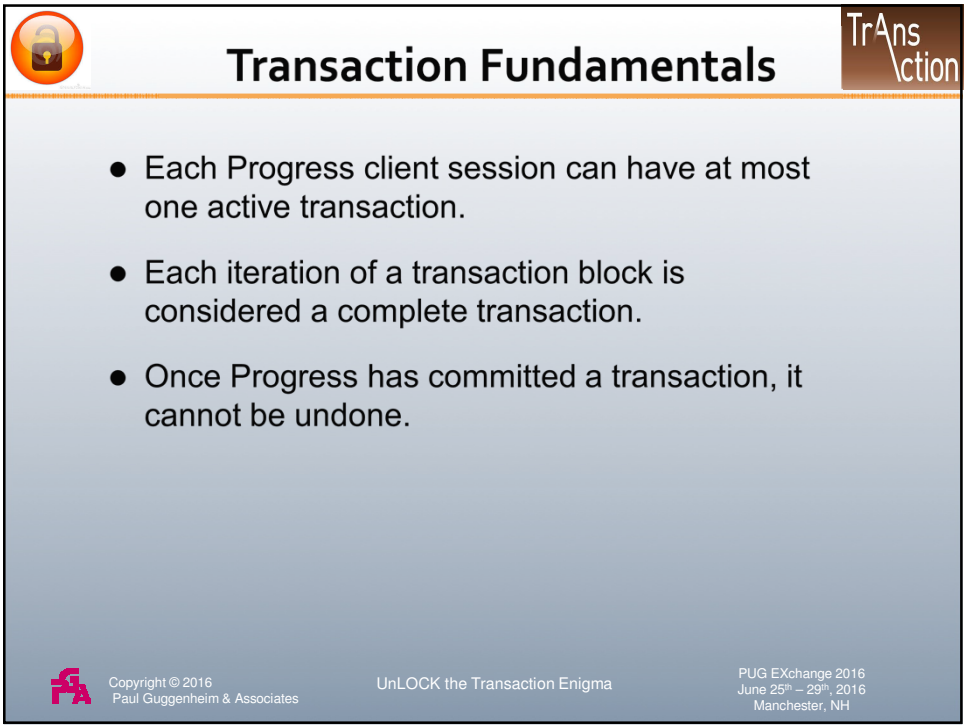
- Transactions
- Transaction Examples
- Record Locks
- Record Scope
- When are locks released?
- Record Locking Examples

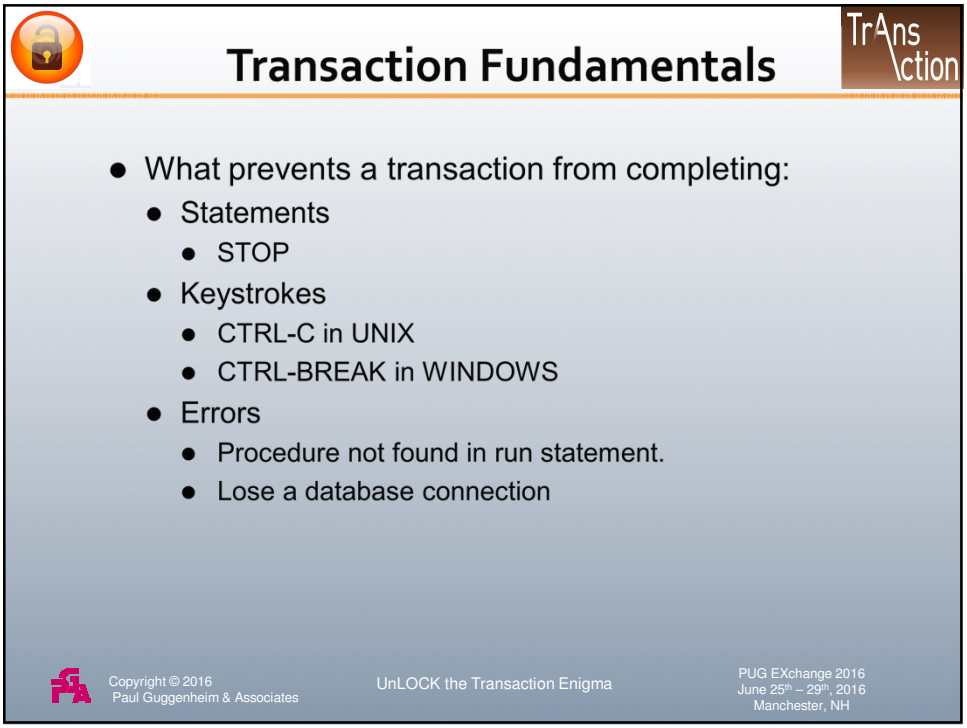
What's a Transaction?

- A transaction is a group of records that must be updated completely or not at all.
  - Transactions are designed to ensure data integrity.
- All transactions are scoped to blocks.
- The default transaction block is the outermost block with transaction properties that contains statements that modify the database.
  - In short, the outermost block that updates the database.


What's a Transaction?
TrAns
ction
What blocks have transaction properties by default?
It's easier to mention which blocks DON'T have transaction properties by default. The DO block.
Put the TRANSACTION keyword on the DO block header to add transaction properties to the DO block.
Containing Procedure, internal procedure, FOR, REPEAT, trigger, method and function blocks all have transaction properties by default.
Override the default transaction block by placing the TRANSACTION keyword on the FOR, REPEAT or DO block.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


What's a Transaction?
TrAns
ction
Statements that modify the database:
ASSIGN
CREATE
DELETE
INSERT
SET
UPDATE
Plus keyword
EXCLUSIVE-LOCK
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

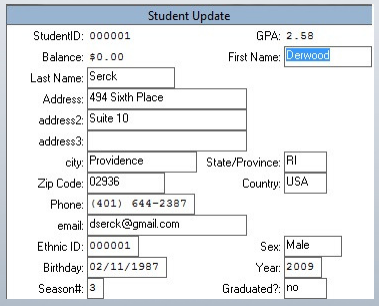

Transaction Fundamentals
TrAns
ction
Each Progress client session can have at most one active transaction.
Each iteration of a transaction block is considered a complete transaction.
Once Progress has committed a transaction, it cannot be undone.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH


Transaction Fundamentals
TrAns
ction
What prevents a transaction from completing:
Statements
STOP
Keystrokes
CTRL-C in UNIX
CTRL-BREAK in WINDOWS
Errors
Procedure not found in run statement.
Lose a database connection
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

Transaction Example One

```
/* trn1.p */
for each student with 2 columns:
  display studentid gpa balanceamt.
  update student
       except studentid gpa balanceamt addressagg picture.
end. /* for each */
```


Transaction Example One



Copyright © 2016
Paul Guggenheim & Associates

UnLOCK the Transaction Enigma

PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

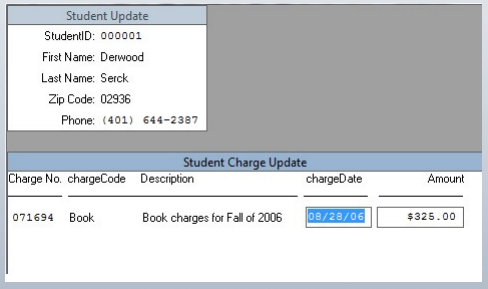
Transaction Example One

- For each block is the transaction block since no changes to database are made in containing procedure block.
- Remember that each iteration of a transaction block is a transaction.

Transaction Example Two


```
/* trn2.p */
for each student:
  display studentid sfirstname slastname.
  update postalcode phone.
  for each stuchrg of student:
    display chargeno chargecode
            studentchargedescription.
    update chargedate chargeamt.
  end. /* for each stuchrg */
end. /* for each student */
```

Transaction Example Two



Transaction Example Two

- When working with related records, it may be desirable to ensure all changes for related records have been made completely or not at all.
  - This is especially true if key values are being changed for related records.
- In this example, the outer **FOR EACH** block is the transaction block.
  - Changes are not committed to the database for each student and its student charges until after the last charge for that student has been updated.
- If the user presses the STOP key (CTRL-BREAK in DOS/Windows, CTRL-C in UNIX) while they are modifying a student or a stuchrg record then all the changes for that student and its charges would be undone.

Copyright © 2016
Paul Guggenheim & Associates

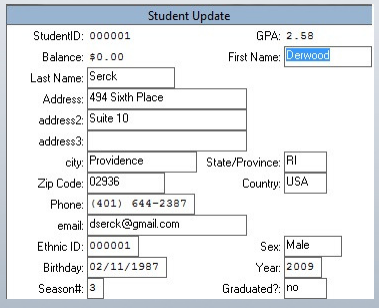
UnLOCK the Transaction Enigma

PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

Transaction Example Three

```
/* trn3.p */
do transaction:
  for each student with 2 columns:
    display studentid gpa balanceamt.
    update student except studentid gpa
           balanceamt addressagg picture.
  end. /* for each */
end. /* do transaction */
```


Transaction Example Three



Copyright © 2016
Paul Guggenheim & Associates

UnLOCK the Transaction Enigma

PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

Transaction Example Three

TrAns\ction

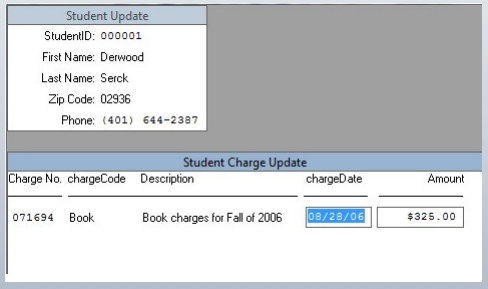
- By using the **DO TRANSACTION** block, you can create larger transactions.
- In this example, the **DO TRANSACTION** block is the transaction block.
  - Changes are not committed for any student until the last student has been updated, since the transaction is not complete until the **DO** block ends.
  - If the transaction block is undone for any reason, none of the changes for any of the students will be committed in the database.

Transaction Example Four

TrAns\ction

```
/* trn4.p */
for each student:
  display studentid sfirstname slastname.
  do transaction:
    update postalcode phone.
  end. /* do transaction */
  for each stuchrg of student transaction:
    display chargeno chargecode studentchargedescription.
    update chargedate chargeamt.
  end. /* for each stuchrg */
end. /* for each student */
```

Transaction Example Four



Transaction Example Four

- The **DO TRANSACTION** block can also be used to create smaller transactions.
  - You may wish to make your transactions smaller when updating related records whose updates are not dependent on each other.
- In example **trn4.p**, the outer **FOR EACH** block is not a transaction block, since it contains no updates to the database that aren't contained within another block with transaction properties.
- The record modifying the student has been explicitly placed within the **DO TRANSACTION** block.
- If the transaction in the inner **FOR EACH** block is undone, only the changes for the last student charge are not committed—the changes for the previous student charges were committed on each iteration of the inner **FOR EACH** block.

Copyright © 2016
Paul Guggenheim & Associates

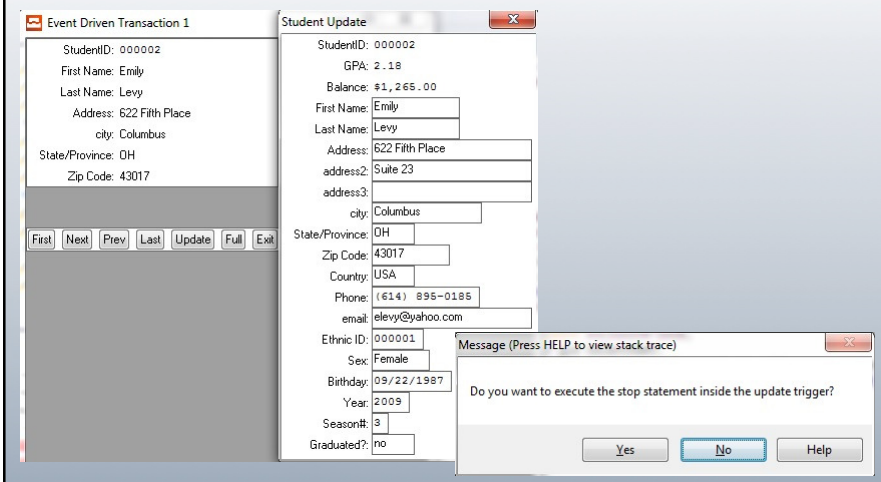
UnLOCK the Transaction Enigma

PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

Transactions and Triggers

```
/* trnevt4.p */

on choose of bupd do:

  if available student then do with 1 column view-as dialog-box title
   "Student Update":

    find  CURRENT student exclusive-lock.

    display studentid gpa balanceamt.

    update student except studentid balanceamt gpa addressagg
   picture.

    message "Do you want to execute the stop statement inside the update
   trigger?" view-as alert-box buttons yes-no update choice.

    if choice then stop.

  end.

end.
```

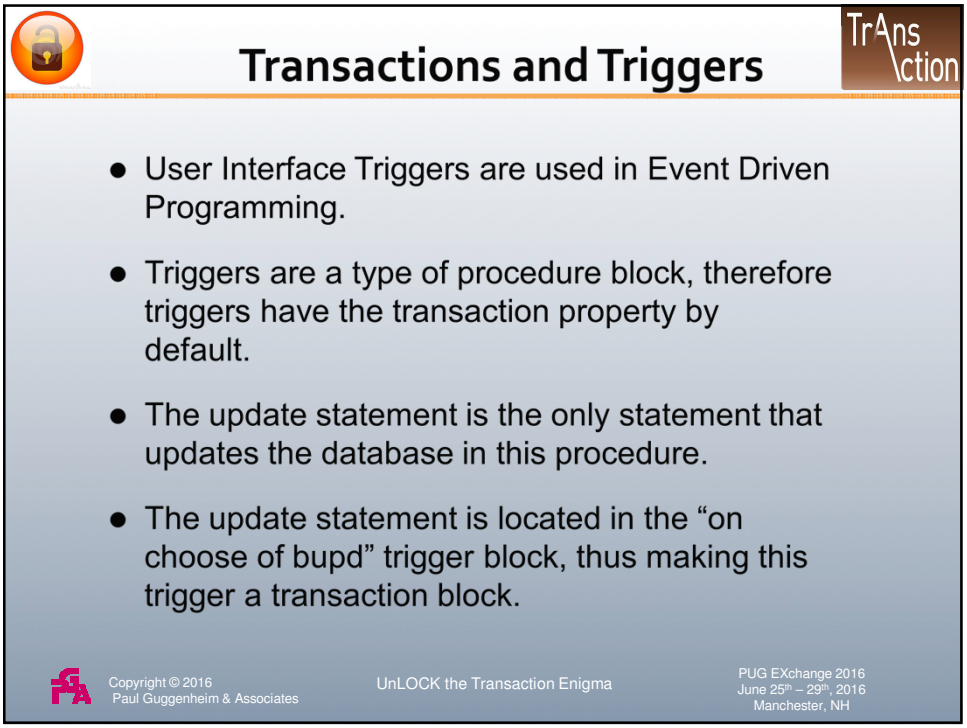

Transactions and Triggers

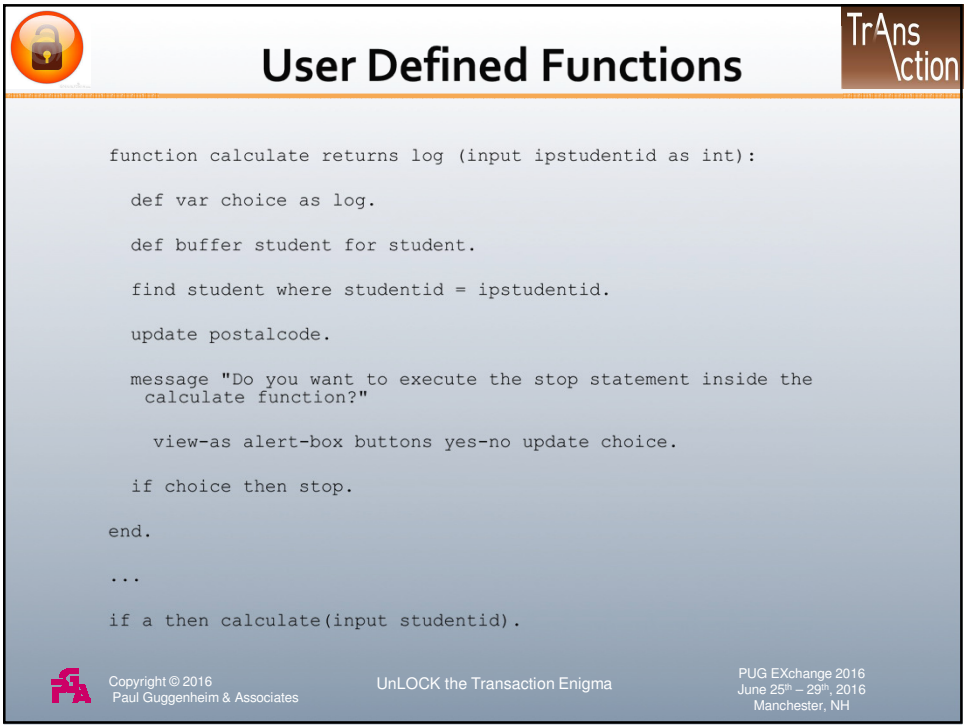


Copyright © 2016
Paul Guggenheim & Associates

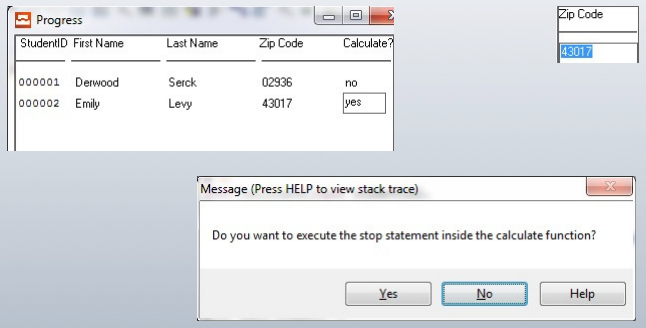
UnLOCK the Transaction Enigma

PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH


Transactions and Triggers
TrAns
ction
User Interface Triggers are used in Event Driven Programming.
Triggers are a type of procedure block, therefore triggers have the transaction property by default.
The update statement is the only statement that updates the database in this procedure.
The update statement is located in the “on choose of bupd” trigger block, thus making this trigger a transaction block.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH


User Defined Functions
TrAns
ction
function calculate returns log (input ipstudentid as int):
def var choice as log.
def buffer student for student.
find student where studentid = ipstudentid.
update postalcode.
message "Do you want to execute the stop statement inside the calculate function?"
view-as alert-box buttons yes-no update choice.
if choice then stop.
end.
...
if a then calculate(input studentid).
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

User Defined Functions



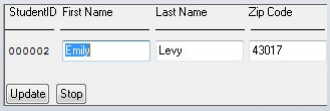
User Defined Functions

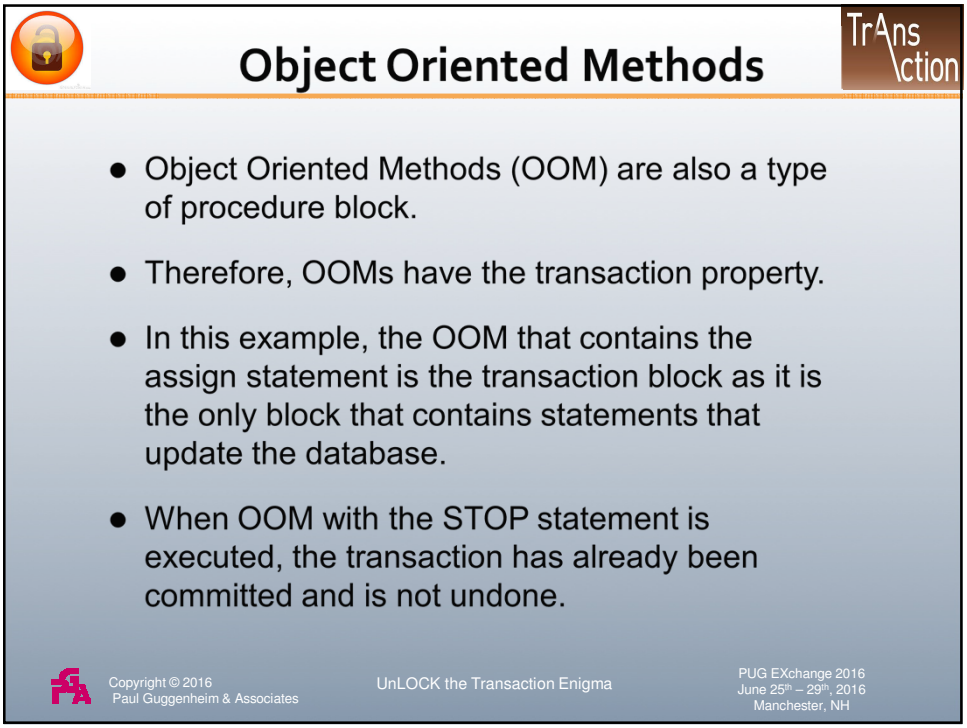
- User Defined Functions (UDF) are also a type of procedure block.
- Therefore, UDFs have the transaction property.
- In this example, the UDF is the transaction block as it is the only block that contains statements that update the database.
- In order to undo the transaction, the STOP statement must be executed in the UDF.

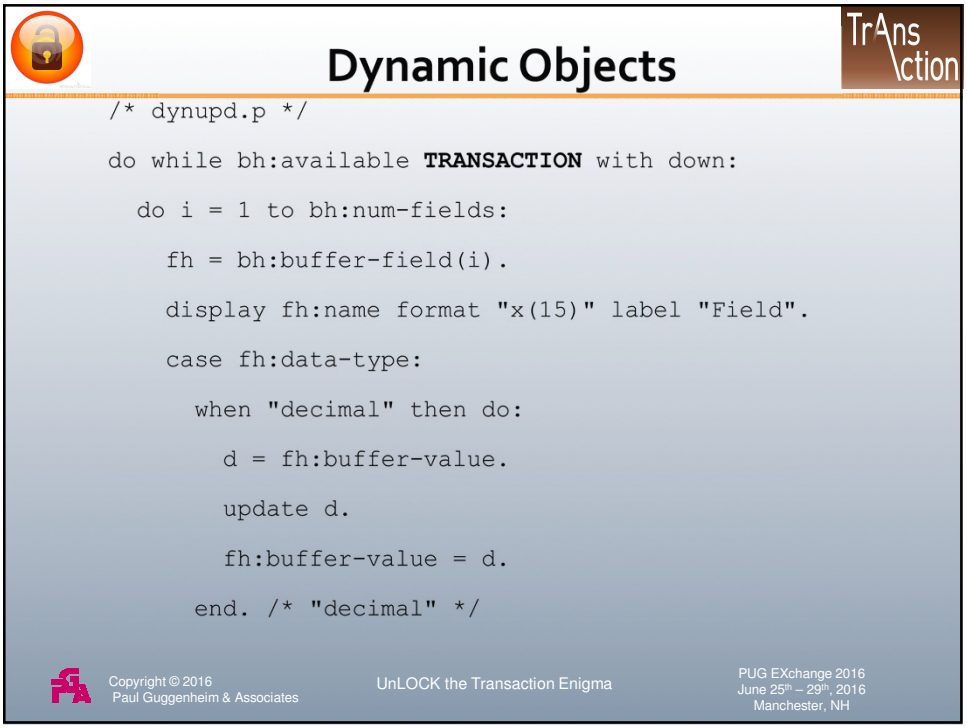
Object Methods

```
/* studenttrans.cls */
...
  METHOD PUBLIC VOID methodupdate(  ):
    do with frame f1:
      assign student.sfirstname slastname postalcode.
    end.
  END METHOD.
  METHOD PUBLIC VOID methodstop(  ):
    stop.
  END METHOD.
```


Object Methods




Object Oriented Methods
TrAns action
Object Oriented Methods (OOM) are also a type of procedure block.
Therefore, OOMs have the transaction property.
In this example, the OOM that contains the assign statement is the transaction block as it is the only block that contains statements that update the database.
When OOM with the STOP statement is executed, the transaction has already been committed and is not undone.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

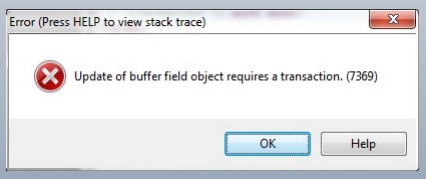

Dynamic Objects
TrAns action
/* dynupd.p */
do while bh:available TRANSACTION with down:
do i = 1 to bh:num-fields:
fh = bh:buffer-field(i).
display fh:name format "x(15)" label "Field".
case fh:data-type:
when "decimal" then do:
d = fh:buffer-value.
update d.
fh:buffer-value = d.
end. /* "decimal" */
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

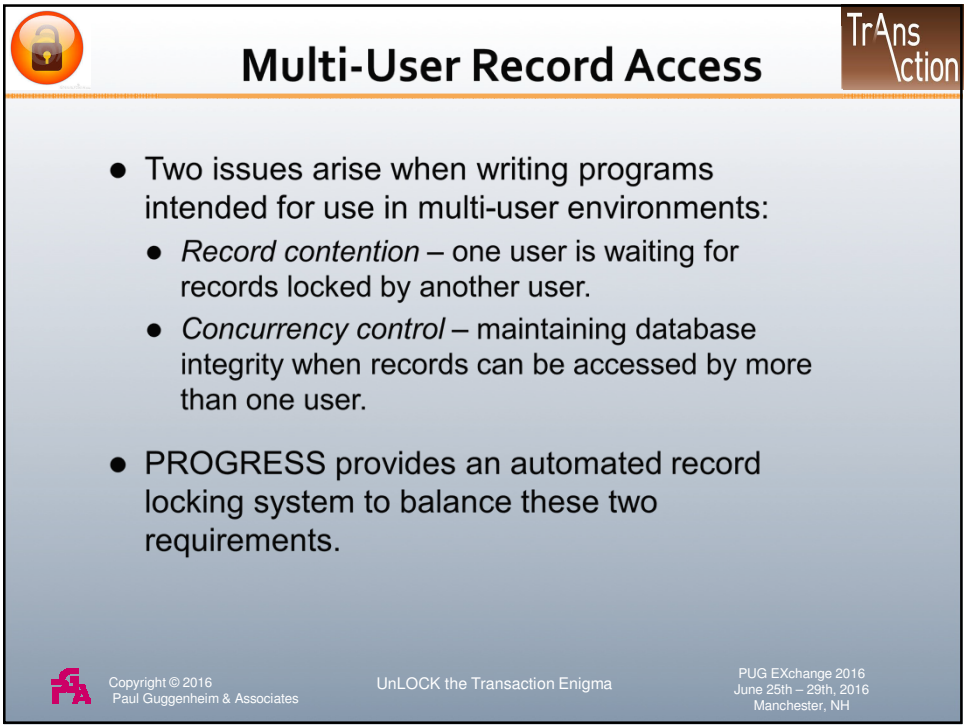
Dynamic Objects

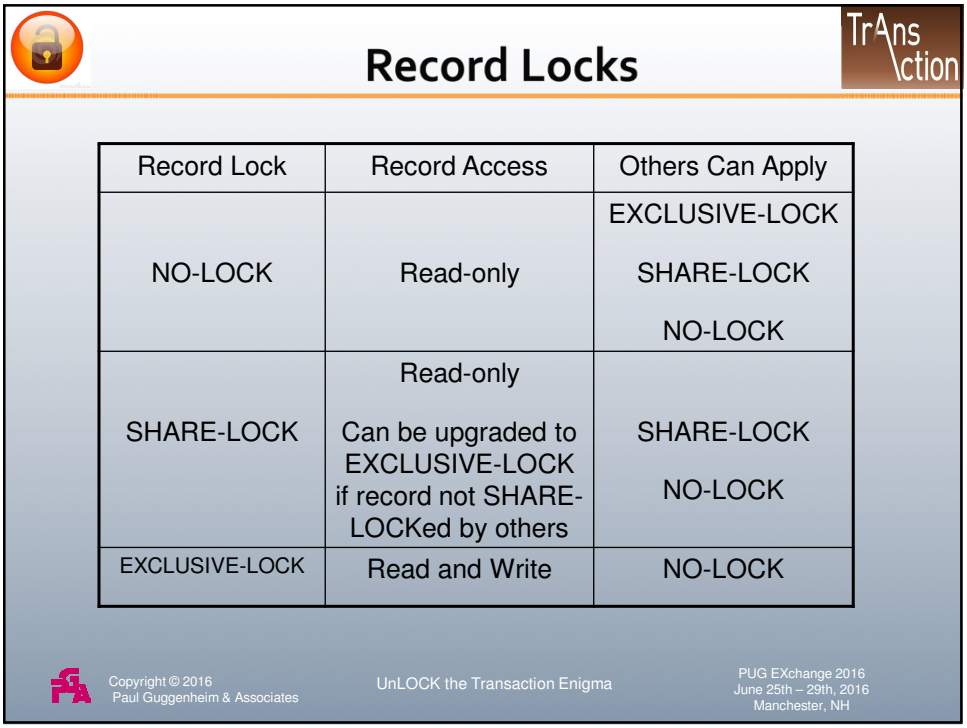
Field	d c
gradePoint	0.00
gradeName	F
gradePoint	0.67
gradeName	D-
gradePoint	1.00

Dynamic Objects

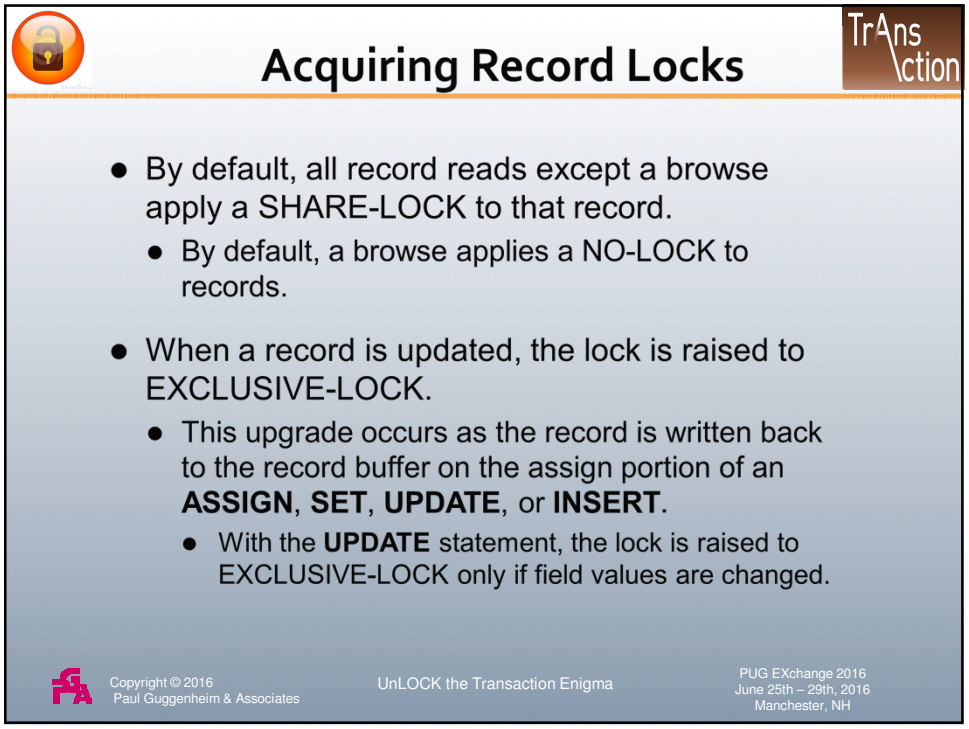
- Because of the flexible and powerful nature of dynamic buffers, queries and fields, Progress requires that the TRANSACTION keyword be specified on the desired transaction block.
- If the TRANSACTION keyword is missing while attempting to update a buffer-field, the following runtime message is displayed:

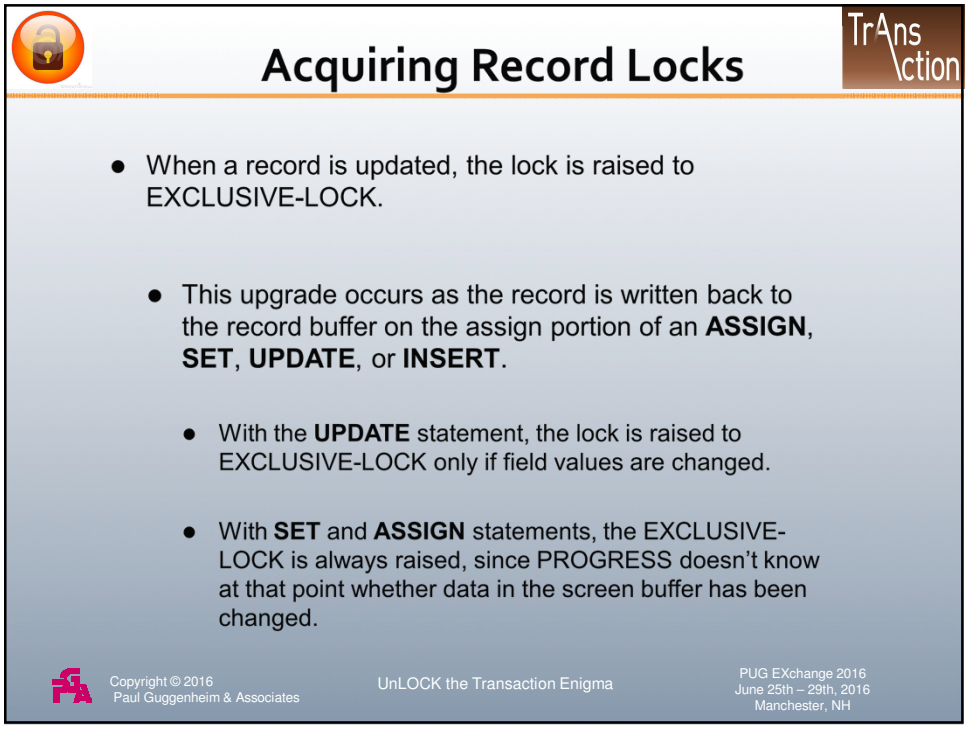


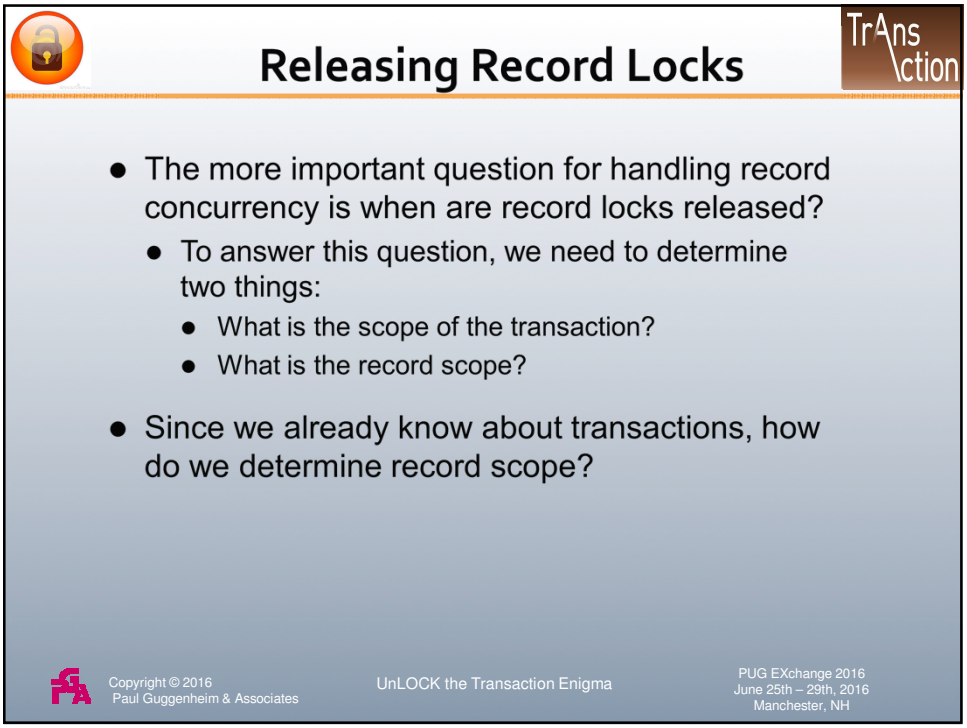


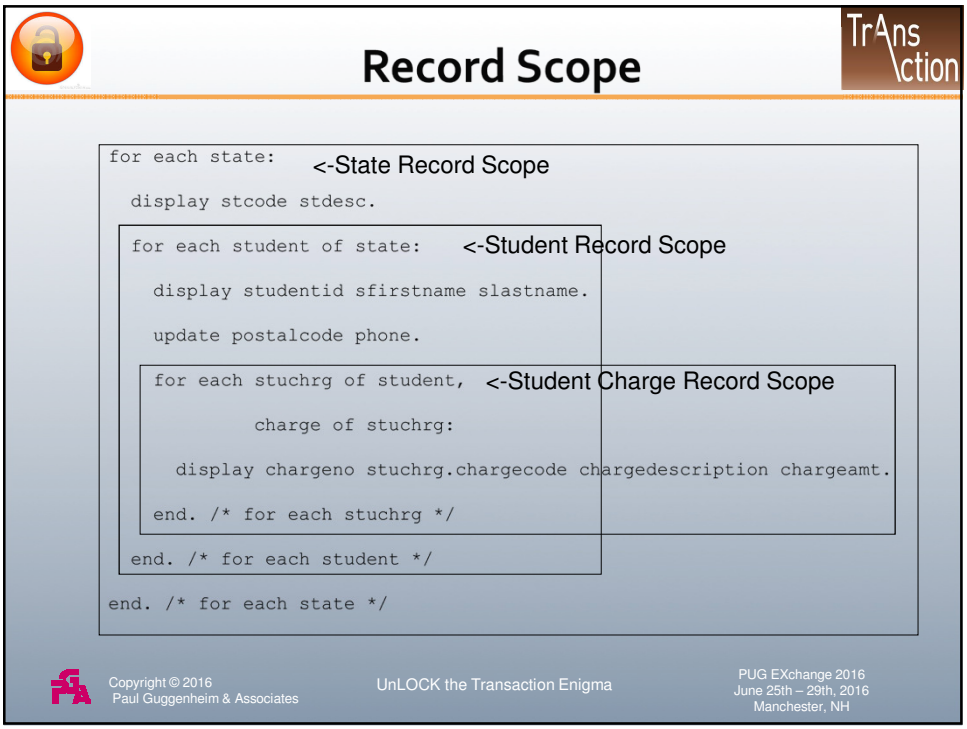


Record Lock	Record Access	Others Can Apply
NO-LOCK	Read-only	EXCLUSIVE-LOCK SHARE-LOCK NO-LOCK
SHARE-LOCK	Read-only Can be upgraded to EXCLUSIVE-LOCK if record not SHARE-LOCKed by others	SHARE-LOCK NO-LOCK
EXCLUSIVE-LOCK	Read and Write	NO-LOCK


Acquiring Record Locks
TrAns action
By default, all record reads except a browse apply a SHARE-LOCK to that record.
By default, a browse applies a NO-LOCK to records.
When a record is updated, the lock is raised to EXCLUSIVE-LOCK.
This upgrade occurs as the record is written back to the record buffer on the assign portion of an ASSIGN, SET, UPDATE, or INSERT.
With the UPDATE statement, the lock is raised to EXCLUSIVE-LOCK only if field values are changed.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


Acquiring Record Locks
TrAns action
When a record is updated, the lock is raised to EXCLUSIVE-LOCK.
This upgrade occurs as the record is written back to the record buffer on the assign portion of an ASSIGN, SET, UPDATE, or INSERT.
With the UPDATE statement, the lock is raised to EXCLUSIVE-LOCK only if field values are changed.
With SET and ASSIGN statements, the EXCLUSIVE-LOCK is always raised, since PROGRESS doesn't know at that point whether data in the screen buffer has been changed.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


Releasing Record Locks
TrAns action
The more important question for handling record concurrency is when are record locks released?
To answer this question, we need to determine two things:
What is the scope of the transaction?
What is the record scope?
Since we already know about transactions, how do we determine record scope?
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

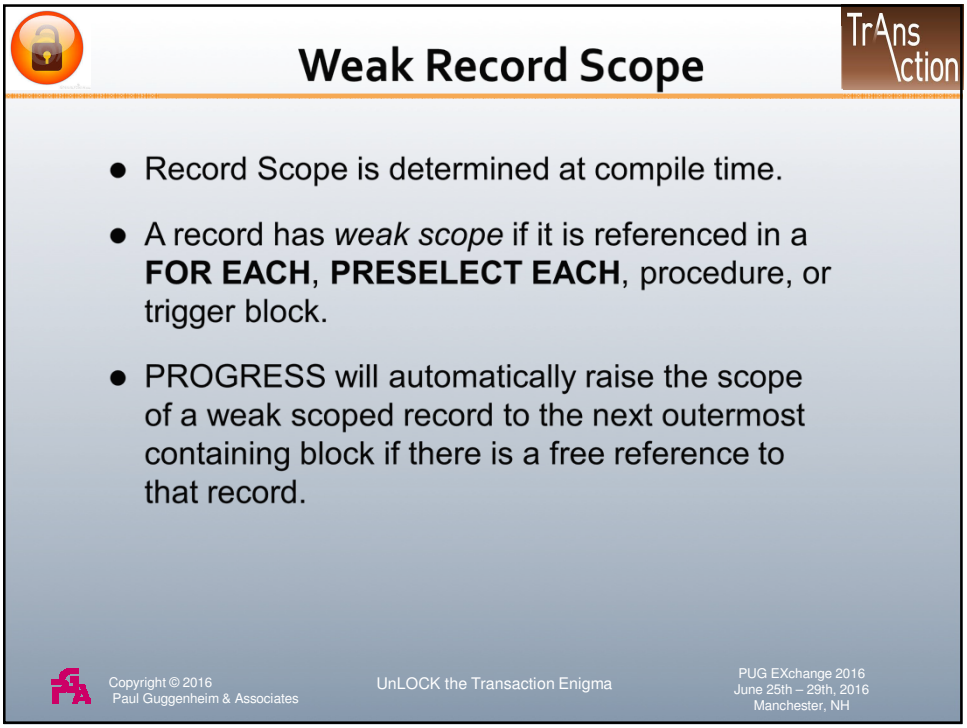

Record Scope
TrAns action
for each state: <-State Record Scope
display stcode stdesc.
for each student of state: <-Student Record Scope
display studentid sfirstname slastname.
update postalcode phone.
for each stuchrg of student, <-Student Charge Record Scope
charge of stuchrg:
display chargeno stuchrg.chargecode chargedescription chargeamt.
end. /* for each stuchrg */
end. /* for each student */
end. /* for each state */
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

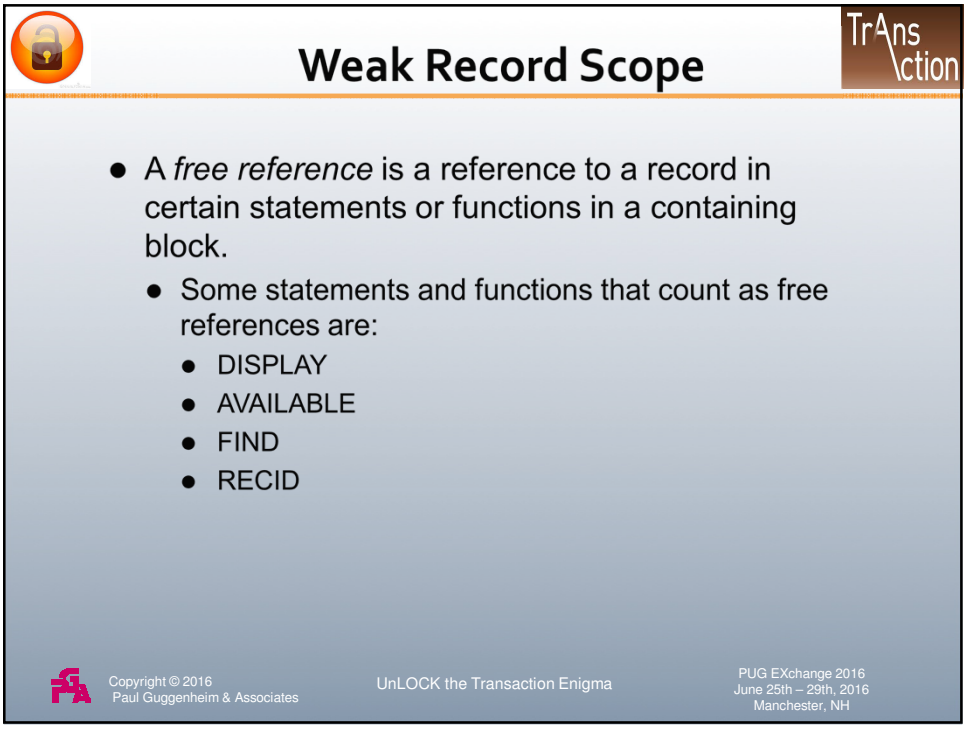
Record Scope

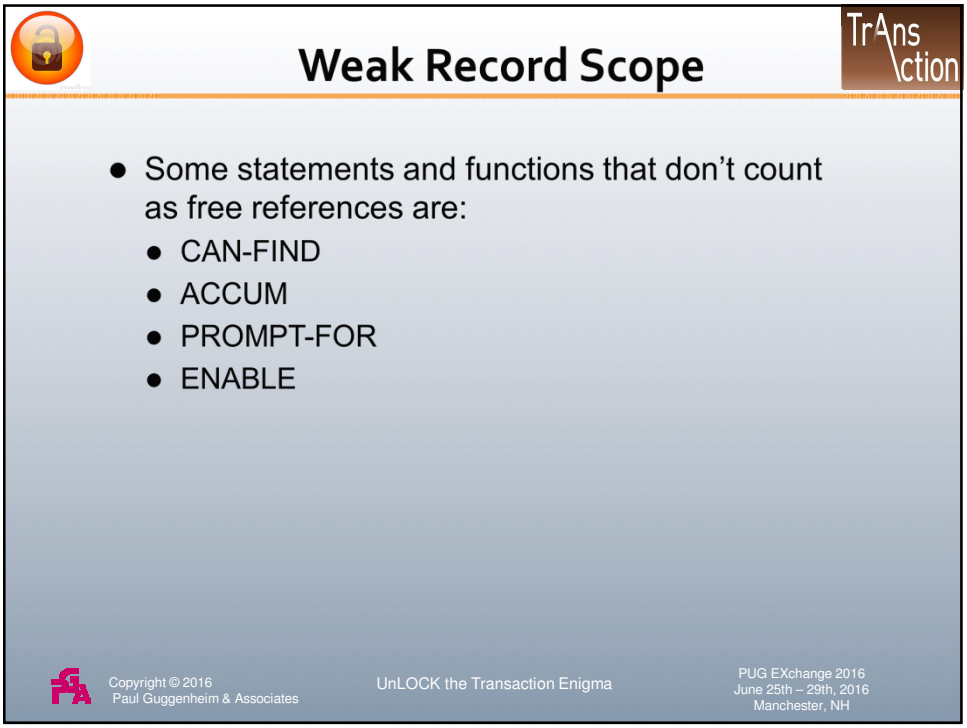
- *Record scope* is the section of a program during which a record buffer exists for a given record.
- In general, a record is scoped to the outermost block that references that record.
- Record scope effects:
  - How long the record is held in the record buffer.
  - When changes to the record are written back to the database.
  - How long record locks are held.

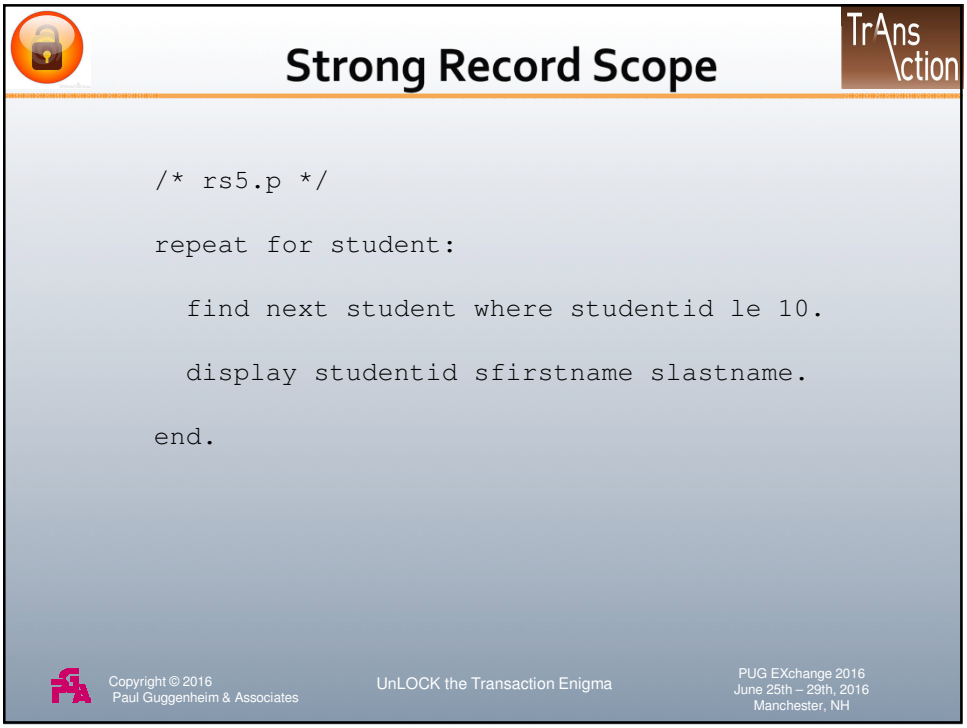
Weak Record Scope

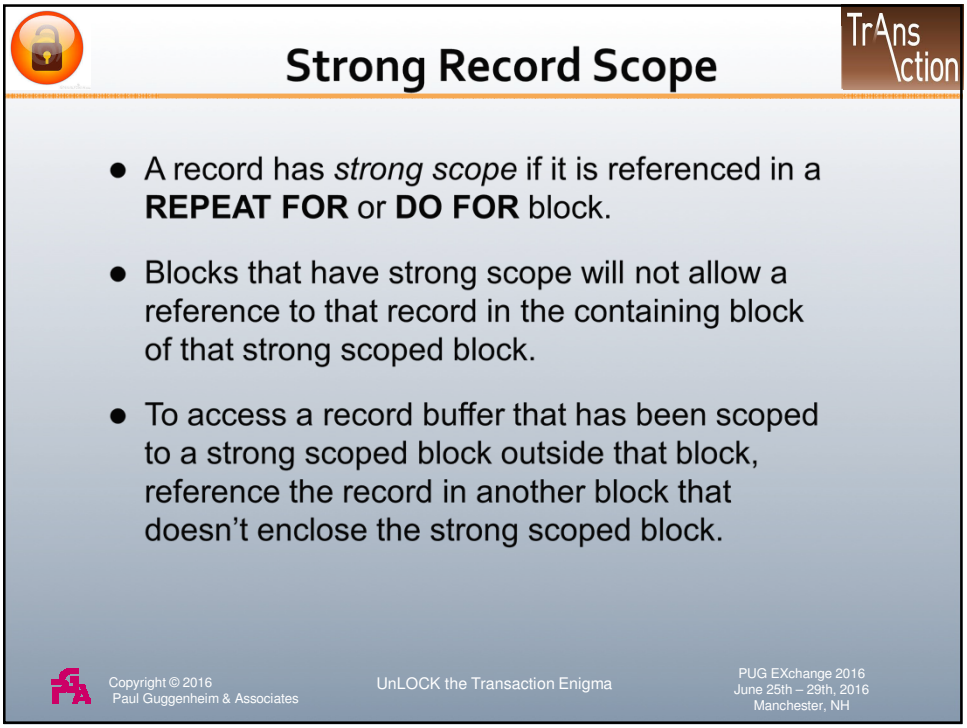

```
/* rs2.p */
for each student:
  display studentid sfirstname slastname.
  if studentid ge 10 then leave.
end. /* for each student */
display sfirstname slastname postalcode studentid.
```

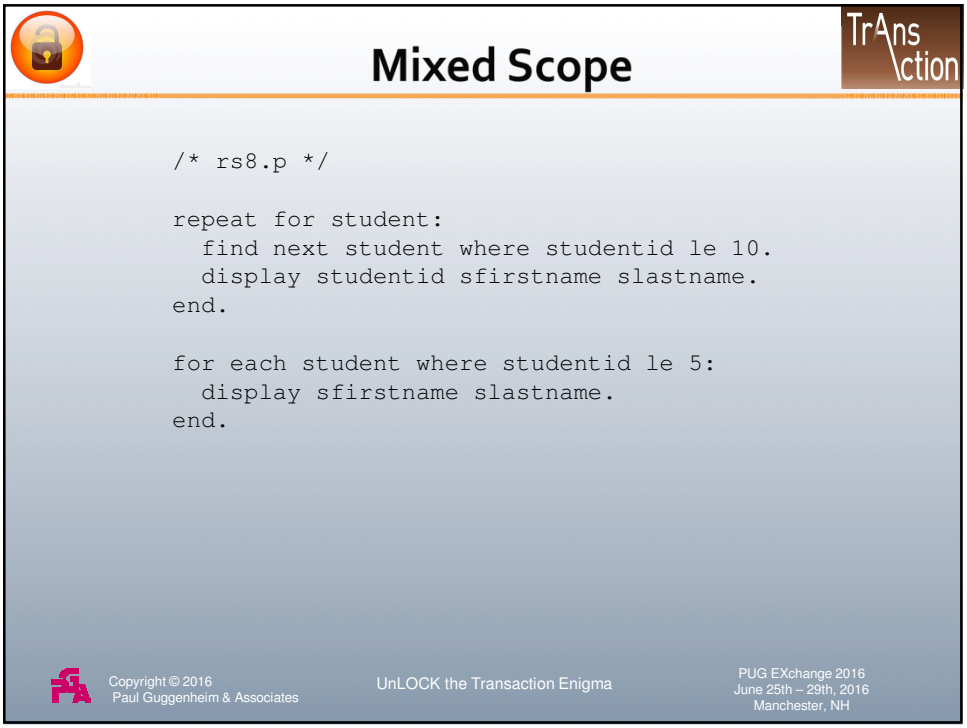

Weak Record Scope
TrAns action
Record Scope is determined at compile time.
A record has *weak scope* if it is referenced in a **FOR EACH**, **PRESELECT EACH**, procedure, or trigger block.
PROGRESS will automatically raise the scope of a weak scoped record to the next outermost containing block if there is a free reference to that record.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

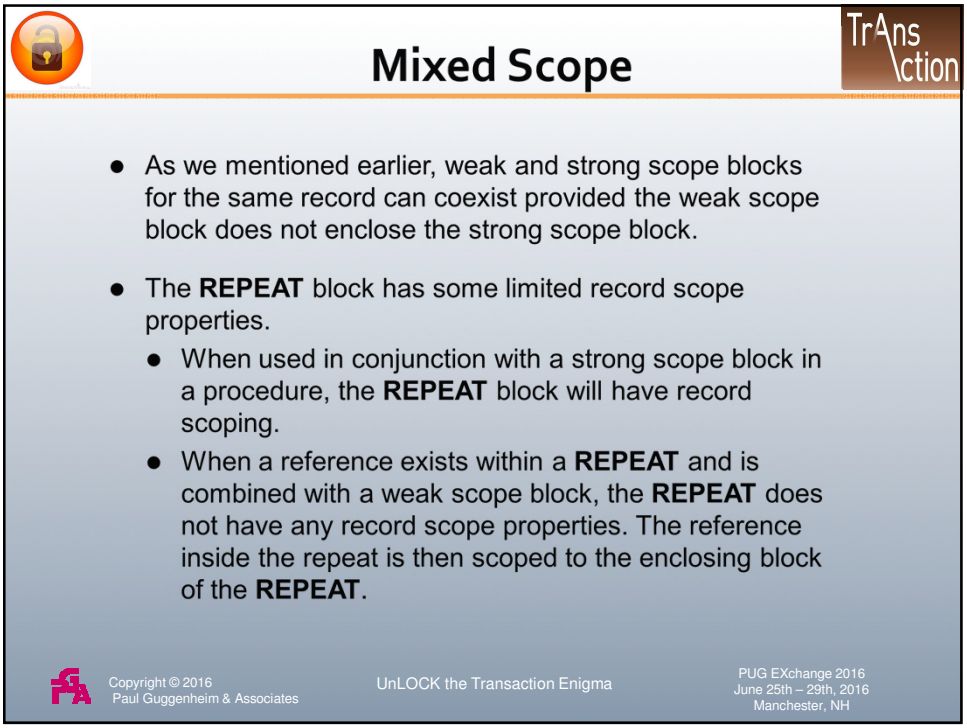

Weak Record Scope
TrAns action
A *free reference* is a reference to a record in certain statements or functions in a containing block.
Some statements and functions that count as free references are:
DISPLAY
AVAILABLE
FIND
RECID
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

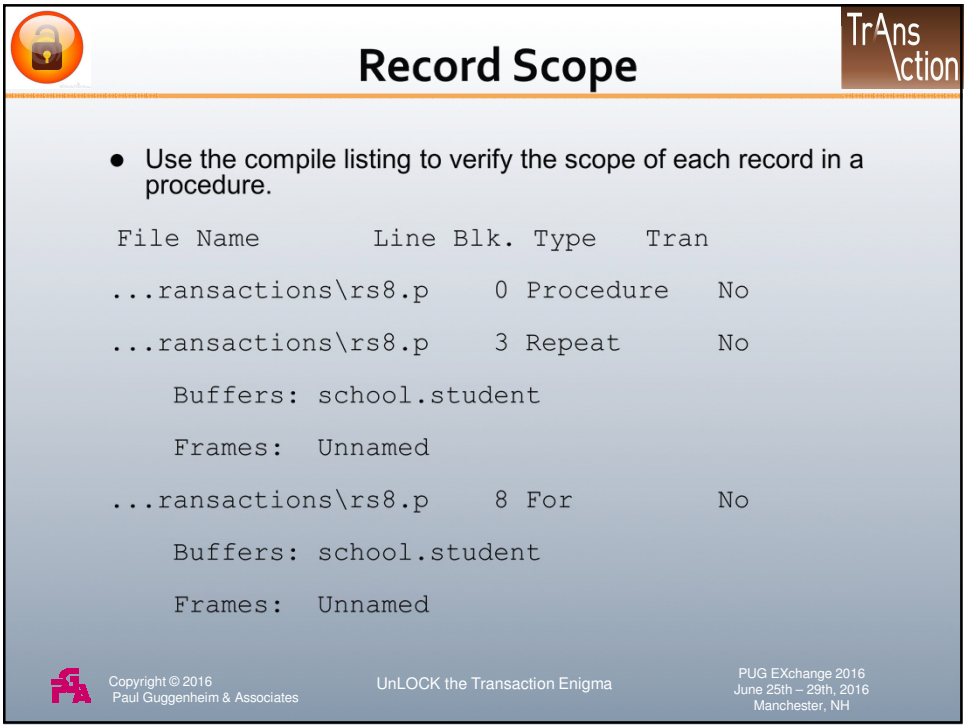

Weak Record Scope
TrAns
ction
Some statements and functions that don't count as free references are:
CAN-FIND
ACCUM
PROMPT-FOR
ENABLE
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH


Strong Record Scope
TrAns
ction
/* rs5.p */
repeat for student:
find next student where studentid le 10.
display studentid sfirstname slastname.
end.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

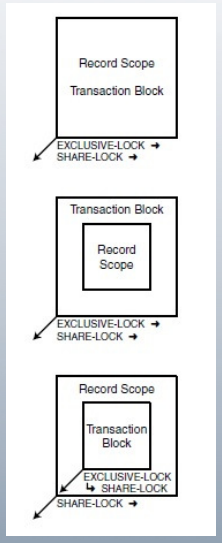

Strong Record Scope
TrAns
ction
A record has *strong scope* if it is referenced in a **REPEAT FOR** or **DO FOR** block.
Blocks that have strong scope will not allow a reference to that record in the containing block of that strong scoped block.
To access a record buffer that has been scoped to a strong scoped block outside that block, reference the record in another block that doesn't enclose the strong scoped block.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


Mixed Scope
TrAns
ction
/* rs8.p */
repeat for student:
find next student where studentid le 10.
display studentid sfirstname slastname.
end.
for each student where studentid le 5:
display sfirstname slastname.
end.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


Mixed Scope
TrAns
ction
As we mentioned earlier, weak and strong scope blocks for the same record can coexist provided the weak scope block does not enclose the strong scope block.
The REPEAT block has some limited record scope properties.
When used in conjunction with a strong scope block in a procedure, the REPEAT block will have record scoping.
When a reference exists within a REPEAT and is combined with a weak scope block, the REPEAT does not have any record scope properties. The reference inside the repeat is then scoped to the enclosing block of the REPEAT.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

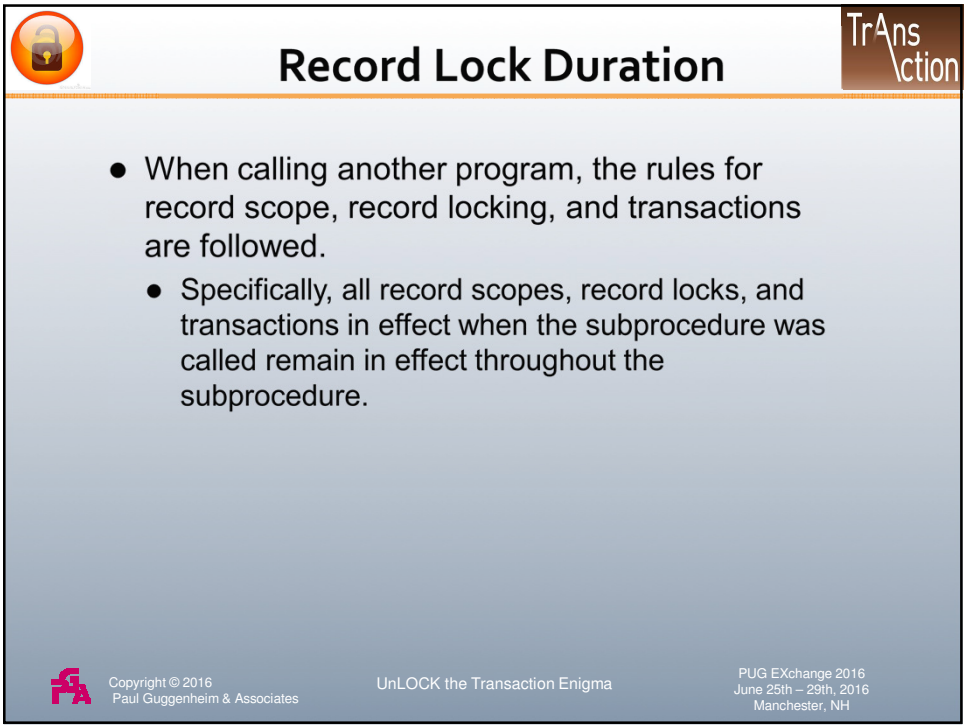

Record Scope
TrAns
ction
Use the compile listing to verify the scope of each record in a procedure.
File Name Line Blk. Type Tran
...ransactions\rs8.p 0 Procedure No
...ransactions\rs8.p 3 Repeat No
Buffers: school.student
Frames: Unnamed
...ransactions\rs8.p 8 For No
Buffers: school.student
Frames: Unnamed
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

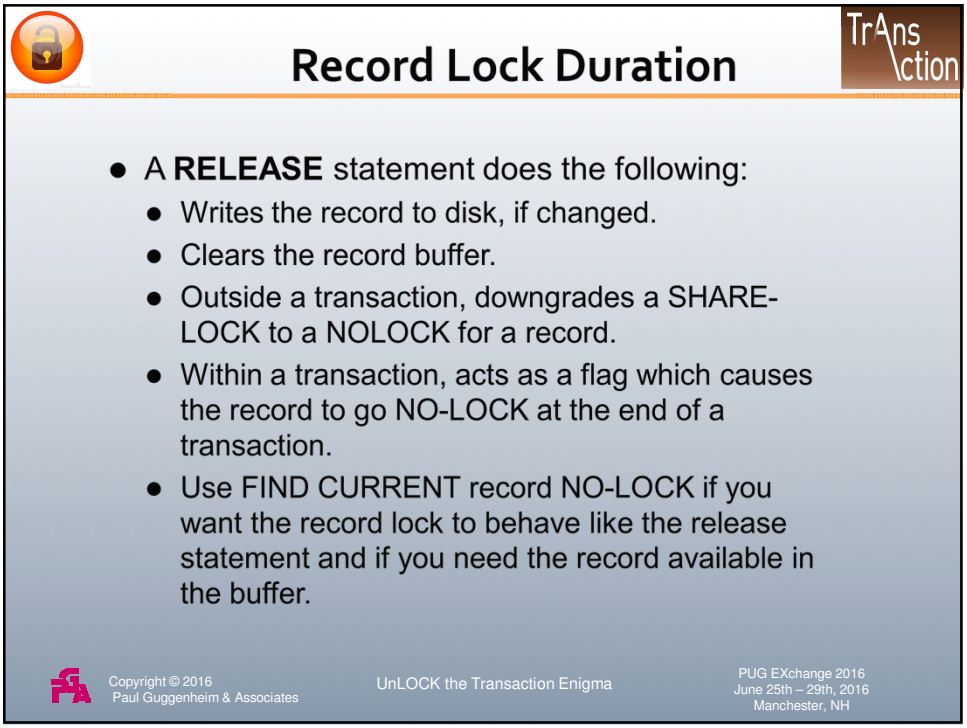
Record Lock Duration

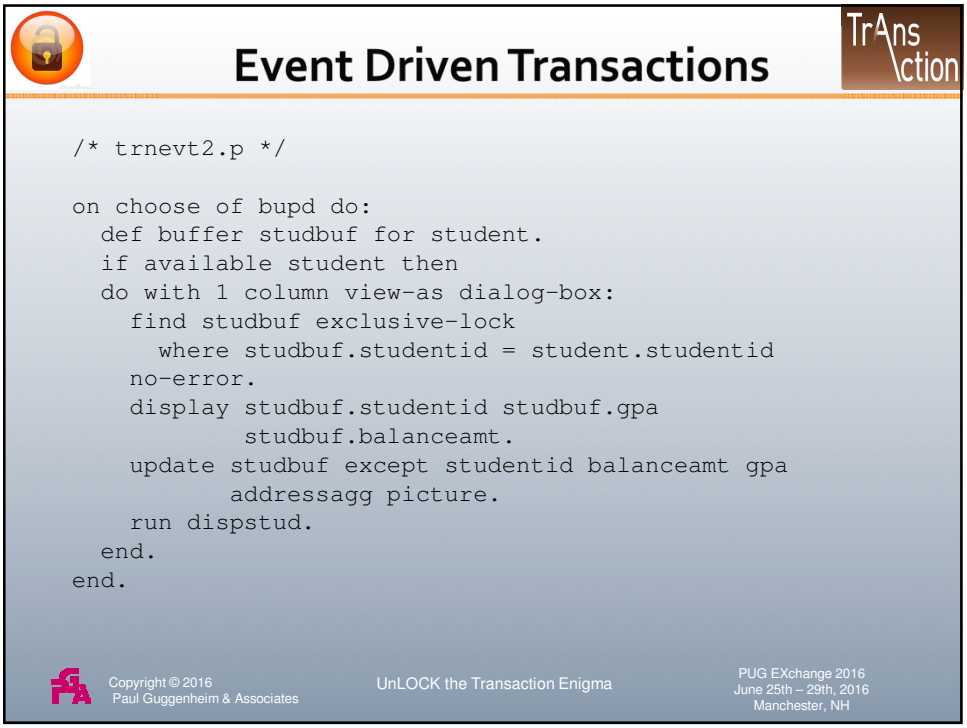


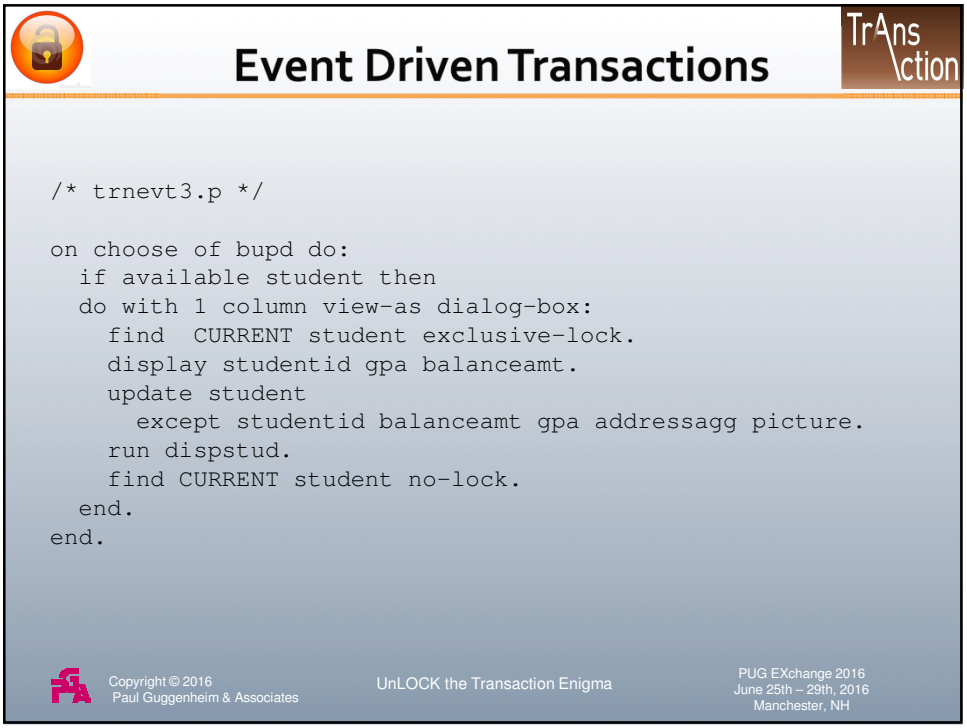
Record Lock Duration

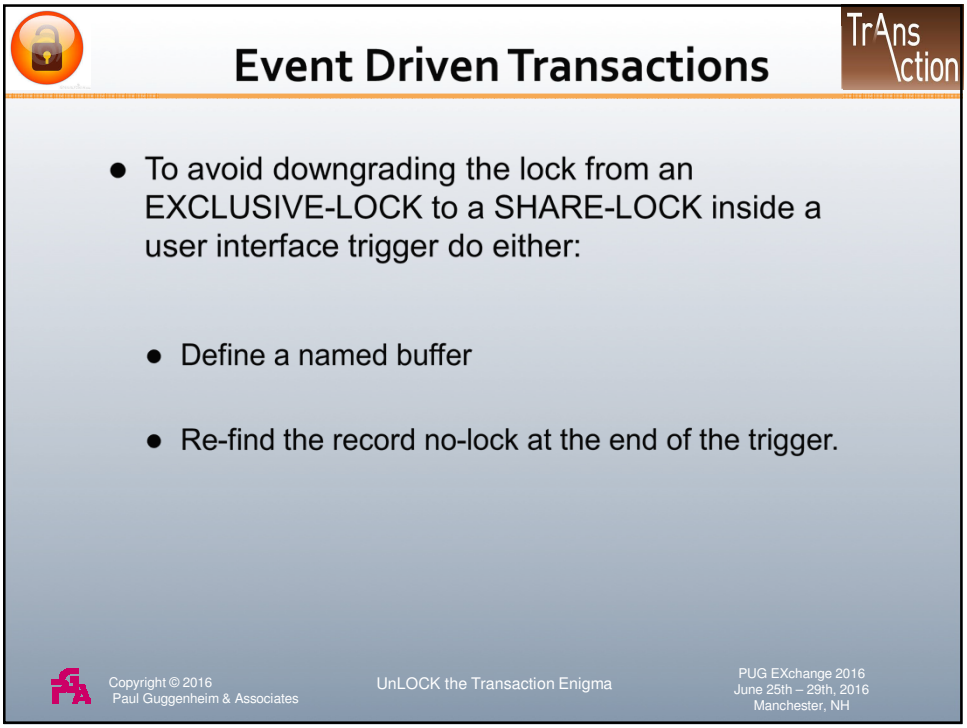
- EXCLUSIVE-LOCKs are held until the end of the transaction block.
- PROGRESS holds SHARE-LOCKs until the end of the block to which they are scoped, or until the end of the transaction, whichever block is larger.
- If the record scope is larger than the transaction block, and the record is EXCLUSIVE-LOCKed in the transaction, at the end of the transaction the record lock downgrades to SHARE-LOCK.

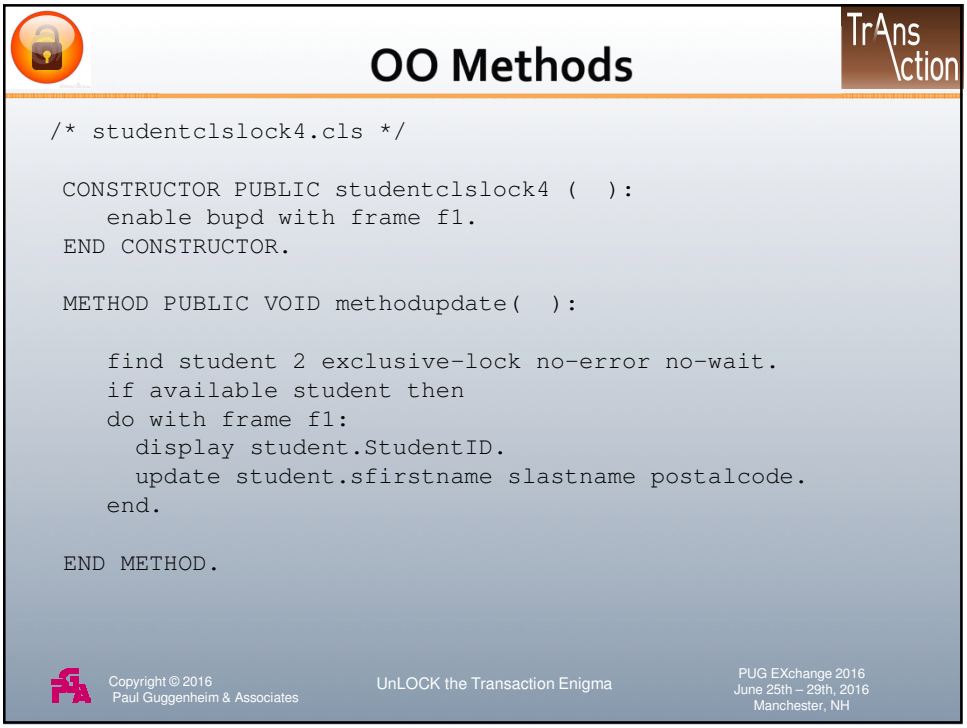

Record Lock Duration
TrAns
ction
When calling another program, the rules for record scope, record locking, and transactions are followed.
Specifically, all record scopes, record locks, and transactions in effect when the subprocedure was called remain in effect throughout the subprocedure.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

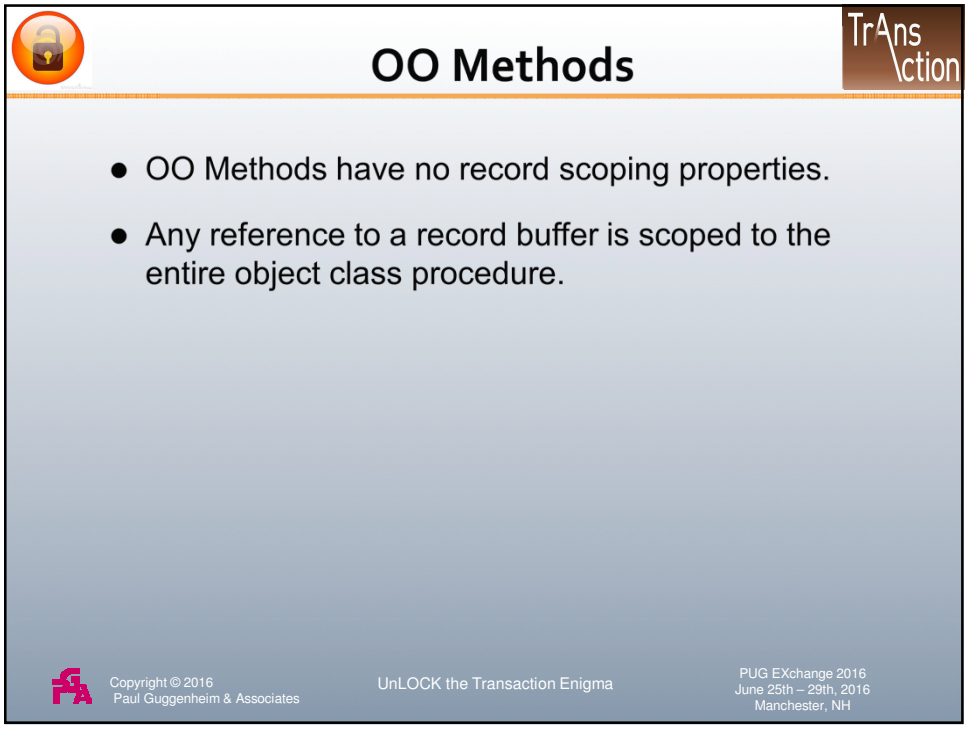

Record Lock Duration
TrAns
ction
A RELEASE statement does the following:
Writes the record to disk, if changed.
Clears the record buffer.
Outside a transaction, downgrades a SHARE-LOCK to a NOLOCK for a record.
Within a transaction, acts as a flag which causes the record to go NO-LOCK at the end of a transaction.
Use FIND CURRENT record NO-LOCK if you want the record lock to behave like the release statement and if you need the record available in the buffer.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH

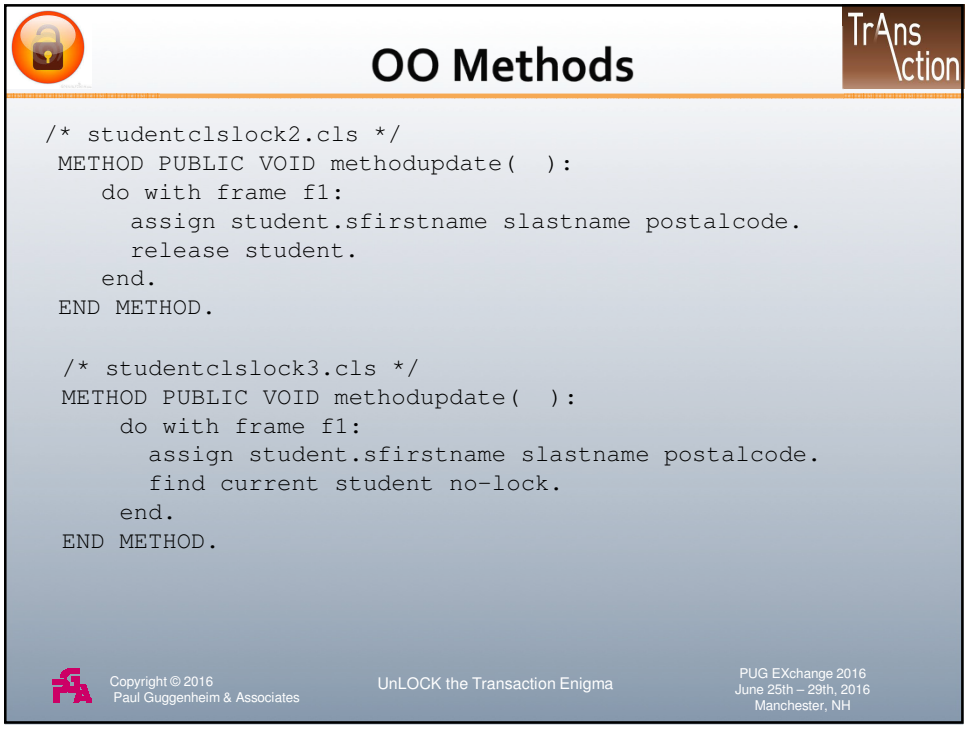

Event Driven Transactions
TrAns
ction
/* trnevt2.p */
on choose of bupd do:
def buffer studbuf for student.
if available student then
do with 1 column view-as dialog-box:
find studbuf exclusive-lock
where studbuf.studentid = student.studentid
no-error.
display studbuf.studentid studbuf.gpa
studbuf.balanceamt.
update studbuf except studentid balanceamt gpa
addressagg picture.
run dispstud.
end.
end.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH


Event Driven Transactions
TrAns
ction
/* trnevt3.p */
on choose of bupd do:
if available student then
do with 1 column view-as dialog-box:
find CURRENT student exclusive-lock.
display studentid gpa balanceamt.
update student
except studentid balanceamt gpa addressagg picture.
run dispstud.
find CURRENT student no-lock.
end.
end.
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH


Event Driven Transactions
TrAns action
To avoid downgrading the lock from an EXCLUSIVE-LOCK to a SHARE-LOCK inside a user interface trigger do either:
Define a named buffer
Re-find the record no-lock at the end of the trigger.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


OO Methods
TrAns action
/* studentclslock4.cls */
CONSTRUCTOR PUBLIC studentclslock4 ( ):
enable bupd with frame f1.
END CONSTRUCTOR.
METHOD PUBLIC VOID methodupdate( ):
find student 2 exclusive-lock no-error no-wait.
if available student then
do with frame f1:
display student.StudentID.
update student.sfirstname slastname postalcode.
end.
END METHOD.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


OO Methods
TrAns action
OO Methods have no record scoping properties.
Any reference to a record buffer is scoped to the entire object class procedure.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


OO Methods
TrAns action
/* studentclslock2.cls */
METHOD PUBLIC VOID methodupdate( ):
do with frame f1:
assign student.sfirstname slastname postalcode.
release student.
end.
END METHOD.
/* studentclslock3.cls */
METHOD PUBLIC VOID methodupdate( ):
do with frame f1:
assign student.sfirstname slastname postalcode.
find current student no-lock.
end.
END METHOD.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH

OO Methods

- Therefore, use the same techniques that are applied for user interface triggers.
- To avoid downgrading the lock from an EXCLUSIVE-LOCK to a SHARE-LOCK inside a OO Method statement do either:
  - Define a named buffer.
  - Re-find the record no-lock at the end of the OO Method block.
- Normally, this isn't the way OO methods should be used, mixing user interface and database update logic. OERA principles should be used.

Deadly Embrace

```
FOR EACH student with 3 down title
                    " User 1 " width 90:
  DISPLAY studentid sfirstname slastname.
  UPDATE postalcode phone.
END.  /* for each student */
```

```
FOR EACH student with 3 down title
                    " User 2 " width 90:
  DISPLAY studentid sfirstname slastname.
  UPDATE postalcode phone.
END.  /* for each student */
```

Deadly Embrace

- When two users read the same record SHARE-LOCK, and then try to update the record to EXCLUSIVE-LOCK, a deadly embrace situation occurs.
- There are three ways to resolve a deadly embrace:
  - Cancel the transaction yourself using the STOP key.
  - Wait for the other person to cancel their transaction.
  - Code procedures to handle locking conflicts more gracefully.

Avoiding the Deadly Embrace


```
/* nowaitold.p */

repeat:
    prompt-for student.studentid.
    find student using studentid exclusive-lock
    no-wait no-error.
    if available student
    then update sfirstname slastname postalcode.
    else if locked student then
       message "Student record is locked, wait until later"
               view-as alert-box.
    else
       message "Student number does not exist"
               view-as alert-box.
end.
```

Avoiding the Deadly Embrace

- The NO-WAIT keyword prevents deadly embraces from occurring on FIND and GET statements.
- NO-WAIT causes the FIND or GET to return control to the procedure instead of the default behavior message of “Record in use by user *user-name* on tty *tty-name*”.
- The logical LOCKED function may be used after the NO-WAIT keyword is used to determine is record is locked.

Batch Program Locking Issues


```
repeat:
  find next student exclusive-lock no-wait no-error.
  if available student then do:
    display studentid sfirstname slastname balanceamt.
    balanceamt = balanceamt * 1.1.
    pause 1.
    display balanceamt.
  end.
  else if locked student then do:
    find next student no-lock.
    display studentid sfirstname slastname balanceamt.
    color display messages studentid sfirstname
                           slastname balanceamt.
    message "student was locked!" view-as alert-box.
  end.
  else leave.
end.
```

Batch Program Locking Issues

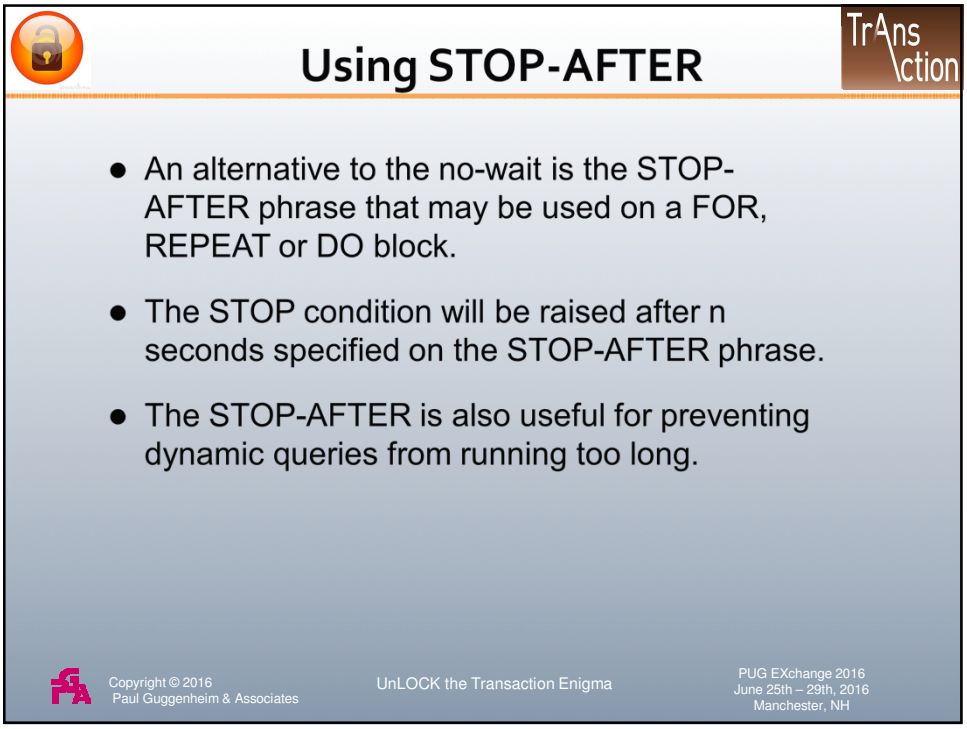
TrAns action

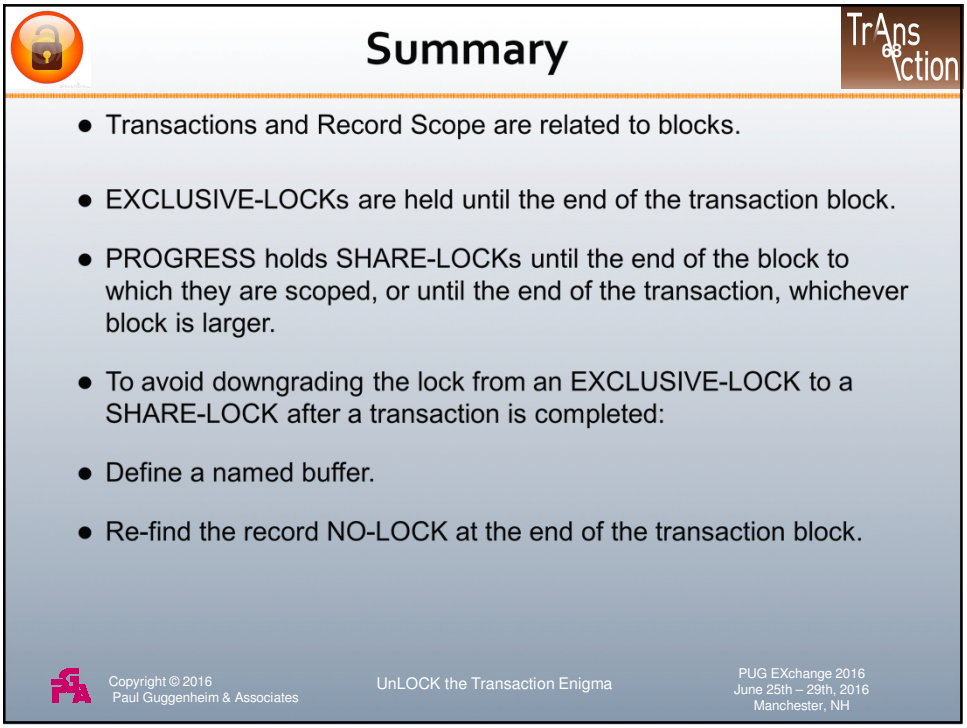
- When updating records in batch, use REPEAT and FIND NEXT instead of FOR EACH so the NO-WAIT keyword can be used.
- If a record is locked, the same record may be found using NO-LOCK since Progress maintains the index cursor in the record scope.
- The ELSE LEAVE statement is necessary since the NO-ERROR on the FIND NEXT suppresses the ENDKEY condition that occurs when a FIND NEXT is attempted after the last record is read.

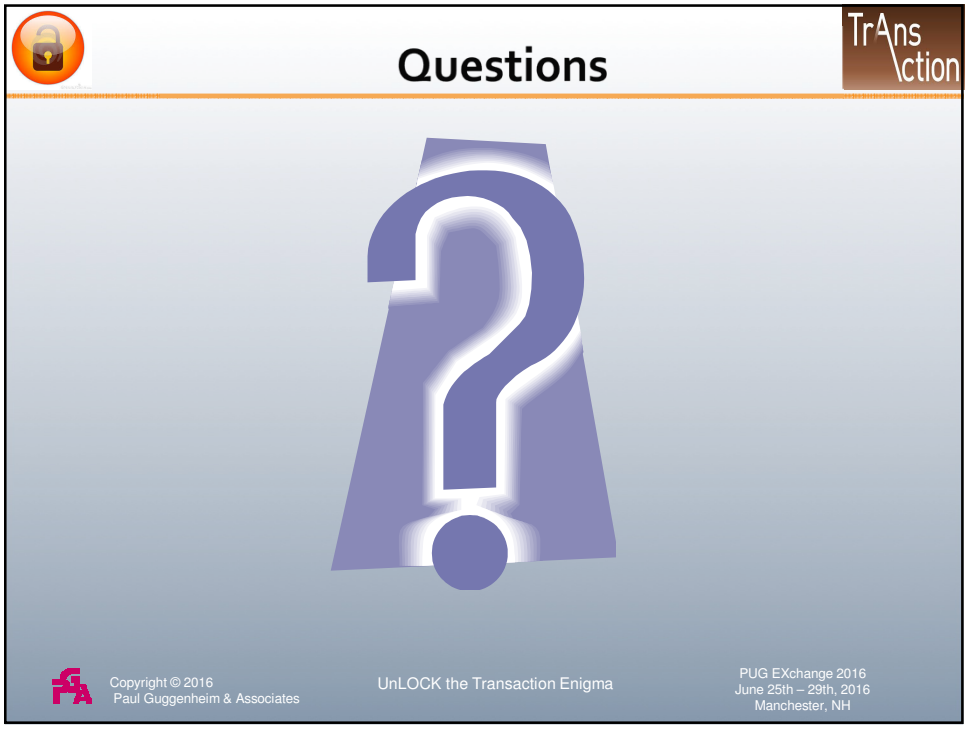
Using STOP-AFTER

TrAns action

```
/* bchupd2.p */

repeat on stop undo, leave stop-after 4:
  find next student exclusive-lock /* no-wait */ no-error.
  if available student then do:
    display studentid sfirstname slastname balanceamt.
    balanceamt = balanceamt * 1.1.
    pause 1.
    display balanceamt.
  end.
  else leave.
end.
display "Program Ended".
```


Using STOP-AFTER
TrAns action
An alternative to the no-wait is the STOP-AFTER phrase that may be used on a FOR, REPEAT or DO block.
The STOP condition will be raised after n seconds specified on the STOP-AFTER phrase.
The STOP-AFTER is also useful for preventing dynamic queries from running too long.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


Summary
TrAns 68 action
Transactions and Record Scope are related to blocks.
EXCLUSIVE-LOCKs are held until the end of the transaction block.
PROGRESS holds SHARE-LOCKs until the end of the block to which they are scoped, or until the end of the transaction, whichever block is larger.
To avoid downgrading the lock from an EXCLUSIVE-LOCK to a SHARE-LOCK after a transaction is completed:
Define a named buffer.
Re-find the record NO-LOCK at the end of the transaction block.
Copyright © 2016 Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016 June 25th – 29th, 2016 Manchester, NH


Questions
TrAns
ction
Copyright © 2016
Paul Guggenheim & Associates
UnLOCK the Transaction Enigma
PUG EXchange 2016
June 25th – 29th, 2016
Manchester, NH