COMP 3311
Database Management Systems

Lab 7

Oracle Indexing
and Oracle PL/SQL Exceptions

Lab Topics

- How to manage an index for a table in Oracle Database.
- PL/SQL exceptions and exception handling.

  Specific information about Oracle indexing and Oracle PL/SQL exceptions can be found by following the links given at the top of some of the slides.

Oracle Database Indexing (1)

http://docs.oracle.com/cd/B28359_01/server.111/b28310/indexes003.htm#ADMIN11722

- An index is used to speed up retrievals, but it can slow down insertion and deletion operations.
- Thus, an index is good for tables used primarily for querying and that do not need to be updated frequently.
- Oracle does not use an index when processing a query in the following scenarios:
  - The select statement does not contain a where clause:

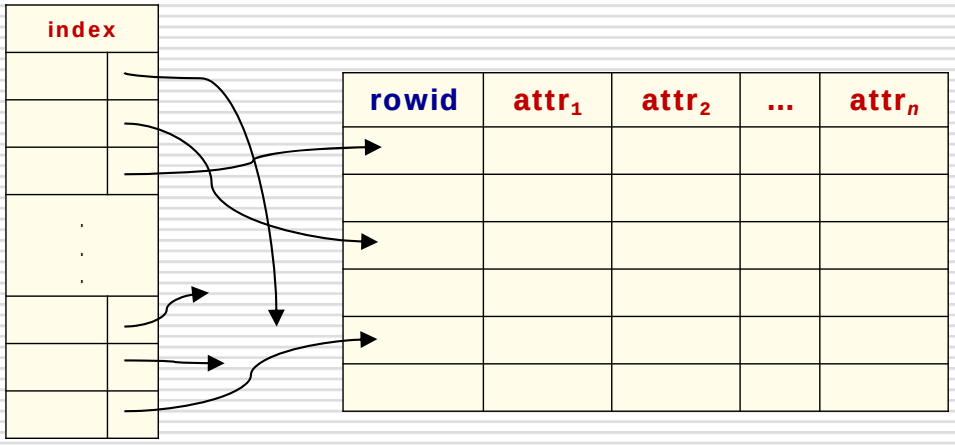
    Example: select * from Course;
  - The select statement contains a where clause, but the where clause does not refer to the indexed attribute(s):

    Example: select * from Course where credits=4;
  - The indexed attribute(s) is/are modified by some function(s) in the where clause:

    Example: select * from Course where substr(courseName, 1, 4)='Data';

Oracle Database Indexing (2)

- Internally, Oracle Database uses a unique attribute known as ROWID to identify records for each table.
- The search key of the index corresponds to the values of the attributes on which the index is created.
- When an index is created, the index entries will hold the values of the search key and the ROWID of the records containing the values of the search key.
- The ROWID information obtained from the index is used to directly retrieve the record.



Managing Indexes

- Create index

  create [unique] index *index_name* on *table_name* (*attribute_name1*, *attribute_name2*, ...);

  Example: Create an index on admissionYear of the Student table.

  create index AdmissionYearIndex on Student (admissionYear);

  Example: Create an index on courseName of the Course table.

  create unique index CourseNameIndex on Course (courseName);

  The unique keyword specifies that the attribute courseName must have unique values.

- Drop index

  drop index *index_name*;

- Display information about an index

  select *index_name* from user_indexes;

RECALL: Oracle PL/SQL Basic Structure

create or replace procedure *procedure_name* [as | is]

Declarative section: declaration of variables, types, and local subprograms go here.

begin **Executable section:** procedural and **SQL** statements go here. This is the only required section of the block.

exception **Exception handling section:** error handling statements go here.

end;

Oracle PL/SQL Exceptions

http://docs.oracle.com/cd/B10501_01/appdev.920/a96624/07_errs.htm

- Predefined exceptions are raised *implicitly* by Oracle PL/SQL if the exception occurs.

  (Refer to the link at the top of this page for a detailed explanation.)

- User-defined exceptions are raised *explicitly* by the command

  raise *exception_name*;

Predefined Exceptions

ACCESS_INTO_NULL	ORA-06530
CASE_NOT_FOUND	ORA-06592
COLLECTION_IS_NULL	ORA-06531
CURSOR_ALREADY_OPEN	ORA-06511
DUP_VAL_ON_INDEX	ORA-00001
INVALID_CURSOR	ORA-01001
INVALID_NUMBER	ORA-01722
LOGIN_DENIED	ORA-01017
NO_DATA_FOUND	ORA-01403
NOT_LOGGED_ON	ORA-01012
PROGRAM_ERROR	ORA-06501
ROWTYPE_MISMATCH	ORA-06504
SELF_IS_NULL	ORA-30625
STORAGE_ERROR	ORA-06500
SUBSCRIPT_BEYOND_COUNT	ORA-06533
SUBSCRIPT_OUTSIDE_LIMIT	ORA-06532
SYS_INVALID_ROWID	ORA-01410
TIMEOUT_ON_RESOURCE	ORA-00051
TOO_MANY_ROWS	ORA-01422
VALUE_ERROR	ORA-06502
ZERO_DIVIDE	ORA-01476

User-defined Exceptions

- Declare the exception in the declaration section.

```
myException exception;
```

- Raise it within a begin...end block.

```
if condition then
    raise myException;
end if;
```

- Define the exception-handling code in the exception section within the begin...end block.

```
exception
    when myException then
        ⋮
```

```
/* DECLARATION SECTION */
myException exception;
        ⋮
begin
    ⋮
  begin
        ⋮
    if condition then
        raise myException;
    endif;
        ⋮
  exception
    when myException then
        /* Code to handle exception */
            ⋮
  end;
        ⋮
end;
```

Continuing After An Exception (1)

- Execution of the block in which an exception is raised terminates after the exception is handled.
- To continue execution after an exception is raised, the statement that can cause the exception must be placed within its own sub-block (i.e., inside its own **begin**...**end** block).
- Execution then resumes after the sub-block in which the exception is raised.

```
/* DECLARATION SECTION */
        :
begin
    :
  begin
        :  ← statements that can cause exceptions are in this sub-block
  exception
    when invalid_cursor then
      /* Code to handle exception */
        :
    when value_error then
      /* Code to handle exception */
        :
  end;
    :  ← execution continues in the outer block after an exception is handled
end;
```

Continuing After An Exception (2)

```
declare
    peRatio number(3,1);
begin
    delete from stats where symbol = 'xyz';
    begin -- sub-block begins
        -- The select statement will throw an exception if nvl(earnings, 0) is zero
        select price / nvl(earnings, 0) into peRatio from stocks where symbol = 'xyz';
    exception
        when zero_divide then
            peRatio := 0;
    end; -- sub-block ends
    insert into stats (symbol, ratio) values ('xyz', peRatio);
exception
    when others then
                :
end;
```

After the zero_divide exception is handled, execution continues with the **insert** statement, which is outside the inner **begin**...**end** block.

Note: The **others** keyword is used to handle any exceptions that are not explicitly named.

Exception Handling Examples

OVERALLAVERAGEGRADEREPORT

Code | Grants | Dependencies | Errors | References | Details | Profiles

Typical Connection

```
1  create or replace procedure OverallAverageGradeReport authid current_user as
2      currentStudentId    Student.studentId%type;
3      sumGrades           number(5,1);
4      avgGrade            EnrollsIn.grade%type;
5      numCourses          int;
6      failingStudent      exception;
7      -- Cursor to fetch Student records
8      cursor studentCursor is select studentId, firstName, lastName from Student order by lastName;
9      -- Cursor to fetch EnrollsIn records for a specified student
10     cursor enrollsInCursor is select grade from EnrollsIn where studentId=currentStudentId;
11 begin
12     for studentRecord in studentCursor loop
13         numCourses := 0;
14         sumGrades := 0;
15         currentStudentId := studentRecord.studentId;
16         begin -- sub-block begins
17             -- Get the number of courses and sum of the grades for a student
18             for enrollsInRecord in enrollsInCursor loop
19                 numCourses := numCourses + 1;
20                 sumGrades := sumGrades + enrollsInRecord.grade;
21             end loop;
22             -- Calculate the average grade
23             avgGrade := sumGrades/numCourses;
24             if avgGrade < 50 then raise failingStudent; end if;
25             dbms_output.put_line(chr(9) || studentRecord.firstName || ' '|| studentRecord.lastName
26                 || '''s overall average grade is ' || avgGrade || '.');
27         exception
28             -- Handle divide by zero error
29             when zero_divide then
30                 dbms_output.put_line('No courses' || chr(9) || studentRecord.firstName || ' '
31                     || studentRecord.lastName || ' is not enrolled in any course.');
32             -- Handle failing student
33             when failingStudent then
34                 dbms_output.put_line('** Failing **' || chr(9) || studentRecord.firstName || ' '
35                     || studentRecord.lastName || '''s overall average grade is ' || avgGrade || '.');
36         end; -- sub-block ends
37     end loop;
38 end OverallAverageGradeReport;
```

1:1

Declaring a user-defined exception

Checking for the user-defined exception

Handling an implicit exception

Handling the user-defined exception