https://www.100test.com/kao_ti2020/143/2021_2022__E5_BC_80_E5_8F_91_E6_8A_80_E6_c102_143236.htm lines 3-6. declare a collection type that mimics the structure of the jokes relational table, and then instantiate a variable based on that type (joke_cache). mU Xyo8aB#*lJA.hL [

http://www.gzu521.com]mU Xyo8aB#*lJA.hL lines 9-11. copy the contents of the jokes table into the cache, using the bulk collect query syntax (the most efficient way to retrieve multiple rows of data into a pl/sql program). note that i am retrieving all the rows in the jokes table. i do not use the limit clause to fetch a subset of the result set's rows, because i have already established that i have sufficient memory to create the larger cache. as you explore places to apply this caching technique in your applications, you should strongly consider using the fetch...bulk collect...limit syntax, so that you can control the size of your pga memory. lines 13-20. use the first and next built-in collection methods to iterate through the collection (result set) from the first row to the last. lines 22-29. use the last and prior methods to go in reverse, from the last row to the first. lines 31-38. compare values in different rows by referencing the index value of the rows in which you are interested. meaningful or cuddly? what are your thoughts about how much information should be displayed to application users when an exception occurs? say a record can't be created because of a primary key constraintshould users see the "ora-00001...," so they can give a meaningful message to support to

help track the problem down, or should all that stuff be hidden, so the user gets a more "cuddly" message, maybe with an error log number pointing to a record in an error log table? my feeling is that when an error occurs, you should provide a message to the user that doesn't violate your company's security protocols, is meaningful, and does not intimidate, in that order of priority. first, let's address the issue of security. when an error occurs in your application or underlying database that you can anticipate and for which you can write explicit handler logic, you can both control the message and tailor it to user needs and the application's business model. if, on the other hand, that error is unexpected, it is quite difficult to know a priori (that is, when you are writing generic error-handling and -reporting code) how much information should be shown. what if the error is caused by a malicious user who is attempting to hack into your web-based application that makes heavy use of dynamic sql? you don't really want to give any information about the state of the database, such as ora-1653 unable to extend table secure_app.personal_info in tablespace sensitive_data.... and you don't want to display "bad" sql statementsstatements made bad by sql injection attempts. you could easily reveal enormous amounts of sensitive information. so for exceptions that are not handled in the pl/sql codeerrors that cannot be anticipated as you write your programsit is best to keep feedback to an absolute minimum, probably something along these lines: an unexpected error has occurred. please contact support and mention the problem identifier 12345. legitimate users will follow up. malicious hackers will look for

softer tar 100Test

www.100test.com