Measuring of Oracle Optimizer Performance for Object-Relational Database Entities

Esam Mahmoud Mohammed
Foundation of Technical Education
Mosul Technical Institute, Iraq

Alaa F.S. AL_Mukhtar
Ministry of Electricity
Northern Electricity, Iraq

Received on: 06/11/2007 **Accepted on: 04/03/2008**

ABSTRACT

SQL Optimization is the process of choosing the most efficient way to execute SQL such as Data manipulation language (Select, Insert, Update, Delete). Relational model represents the object through the use of relations; on the other hand the main aim of ORDBMS model is to represent the special characteristics of OOP. The first model deals with the Relation & Correlated sub queries, while the second one uses collection types (Object type, Varrays, Nested table). Explain plane is used to examine exactly how oracle execute SQL statements " oracle SQL analyze" provides a facility for easily generate any explain plan . DML language (SQL) as tools for any comparison can be achieved by the statistical performance parameter (Elapsed time, CPU time, Logical block read, Physical block read …). Statistical analysis using ANOVA TABLE is used to judge between comparisons. We found that nested table was the most effective type of data, the object type was the least effective (consume lowest time of execution). We also found that increasing data size will increase any of performance data parameter until we reach size of 40% the time begins to decrease. It is found that SELECT, UPDATE and DELETE have the most influence on the performance parameter, respectively.

Keywords: Oracle, Database.

قياس أداء أوراكل الأمثل لقواعد البيانات الكياتية العلائقية

عصام محمود محمد
المعهد التقني/هيئة التعليم التقني/الموصل

الاء المختار
الكهرباء الشمالية/وزارة الكهرباء

تاريخ الاستلام: 2007/11/06 **تاريخ القبول: 2008/03/04**

الملخص

إن الامثلية هي عملية اختيار أكفا الطرائق في تنفيذ ايعازات لغة معالجة البيانات SQL . يتم تمثيل الكائنات في النموذج العلائقي من خلال استعمال العلاقات ، بينما الهدف الرئيسي من استخدام نموذج قواعد البيانات الكياتية العلائقية ORDBMS هو لإظهار الصفات الخاصة

بالبرمجة الشيئية .إن النموذج الأول يتعامل مع العلاقات والاستفسارات الثانوية المرتبطة، أما النموذج الثاني فيستخدم أنواع الكائنات ومصفوفات المتغيرات والجداول المتداخلة. ويمكن تقييم لغة معالجة البيانات المتمثلة بلغة الاستفسارات المهيكلة تقييما إحصائيا بوصفها أداة من خلال المعلمات آلاتية (الوقت المستغرق في التنفيذ، وقت تنفيذ وحدة المعالج المركزي، القراءة المنطقية والقراءة الفيزيائية للكتل البيانية....). إن التحليل الإحصائي باستخدام جدول تحليل التباين يمكننا من الحكم على عمليات المقارنة . كانت الجداول المتداخلة هي الأكثر استهلاكا للوقت والمساحة قياسا بانواع البيانات الاخرى . وقد تبين ان زيادة حجم البيانات يزيد أي من قيم المعلمات لغاية الوصول إلى الحجم 40% حينها تبدأ عملية التناقص . وقد وجد ان الايعازات SELECT, UPDATE, DELETE هي الأكثر تأثيرا في أدائها على التوالي.

الكلمات المفتاحية: اوراكل، قواعد البيانات.

1. Introduction

Much of what is needed to build large scale distributed information systems is already in existing relational systems and their capabilities continue to grow at an exciting rate. Rather than starting a new and risk never catching the moving relational "train".On board and extend the technology in a pragmatic way to realize the important benefits of object technology. The following sections will look at how the familiar relational model can be extended in an evolutionary way so that both technologies can work together in the same server. Users of the system can build existing skills and techniques to store, access and update object data together with relational data in the same system using the same schema with new and current tools [2]. Object functionality is one of the major features of Oracle8; however, relatively little has been written about the performance impact of its use [4].

Relational model represents the object by using relations. While the main aim of ORDBMS model is to represent the special characteristics of OOP(Object Oriented Programming) of Database without looking to the Relational characteristics that are used with the present application ORDBMS an more richer than RDBMS model ,and this richness leads to consistency and good structure in designing database of complex case like data collection [6].

A comparison of several queries against paired relational tables and nested tables found that data access was almost equal in terms of

performance. The only significant problem was a functional retrieval from the detail table for a given master row. In summary, leveraging the full power of some of these features will only be realized by moving wholly to an OO approach. As for object collections, nested tables will not bring major benefits unless one is using object tables [4].

A case study using typical master-detail type data from a live system was undertaken to Investigate two of the most useful features object type and object collections. Nested Table object collections may be useful for implementing master-detail type relationships. A limiting factor in terms of functionality is that access to the detail table is solely through the master table, so look-ups from the detail table are not allowed.

2. Procedure and Methods

The Oracle 9i database was installed with 256MB of RAM running MS Windows XP. In order to study the new object-related features in oracle 9i, case studies were performed against data from a production system. The chosen data comprises a suitable number of relational customer & address records –forming a typical master-detail relationship. Several queries and DML statements were used as shown below.

Data of 600000 records where created for table customer and it's related table of address using program .Also data types of the same size with many levels of runs where done to achieve statistical requirements using different kinds of DML statements ,with recording performance parameters with different sizes of data.

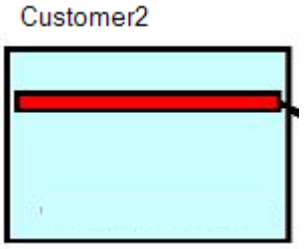
SAS computer package were used for statistical Analysis of the performance statistic data cost (millisecond), Elapsed time this data item represents the total amount of time spent parsing SQL statements (millisecond). Besides amount of time spent by CPU is the number of disc accesses affect the query processing efficiency. Logical blocks read is to access and process those data through memory (No.) .Physical blocks read is transferring data from disk to memory (No.). UGA (Byte), or user global area, is allocated in the PGA (Program global area in Byte) for each session connected to oracle in a dedicated server environment which is measured in three replications by using statistical page of SQL ANALYZER of ORACLE 9i .ANOVA statistical analysis where used to give Duncan's multiple range test which used to distinguish between the means of data by giving them letters [11].

2.1 Relation table

The foundation of Relational Database Management Systems (RDBMS) is the relational model first described by E.F .Codd in 1970 .The success of this technology is largely due to the elegant simplicity of the relational model. Which nevertheless is capable of supporting large scale systems, which are robust and versatile [3].

Creation of simple Relational table as shown below in figure(1):-

```
CREATE TABLE customer2
(customer_id       varchar2(5),
 Last_name         varchar2(20),
 first_name        varchar2(14),
 street            varchar2(120),
 city              varchar2(30),
 state             char(2),
 zip               number(5));
```



figure(1): relation table

```
INSERT  INTO customer2
(customer_id,  last_name ,  first_name  ,  street  ,  city   ,  state   ,  zip);
VALUES(  '22222'  ,  'Smith'  ,  'Jonny'  ,  'S. Main'  ,'Moscow',   'ID' , 43);
```

2.2 Correlated sub queries

A correlated subquery is one that the subquery refers to values in the parent query . A correlated subquery can return the same result as a join, but can be used where a join cannot, such as in an update,insert and delete statement. In a correlated subquery , the subquery executes repeatedly, once for each value of a candidate row selected by the main query, this is why a correlated update can take more processing time[12].

Creation of one to many Relational table as shown below in figure(2):-

```
CREATE TABLE  new_agentr
 (customer_id       varchar2(5),
   last_name        varchar2(20),
   first_name       varchar2(14));
```

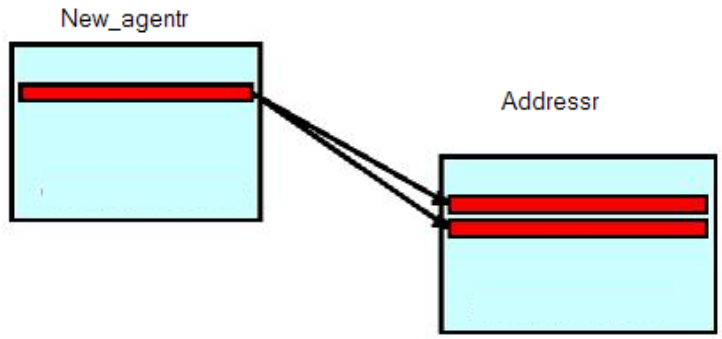


figure (2): correlated sub queries

```
CREATE TABLE addressr
(customer_id        varchar2(5),
 street             varchar2(120),
 city               varchar2(30),
 state              char(2),
 zip                number(5));


INSERT INTO new_agentr
          ( customer_id , last_name , first_name );
VALUES (  '11111'       ,  'Smith'  ,  'wood'    );

 INSERT INTO addressr
          ( customer_id ,  street        , city     ,  state ,  zip     );
 VALUES (  '11111'     , '1190 S. Bend' , 'Pullman' , 'WA'  ,  '98119' );
```

2.3 Object Types

Object types are used to extend the modeling capabilities provided by the native data types. Object types can be used to make better models of complex entities in the real world by binding data attributes to semantic behaviors.[5]

Abstract Data Types (ADTs) from the SQL3 standards work will be introduced as the basic representation mechanism for objects. Our ADTs have the following properties:

One or more attributes. Zero or more methods (methods can be

expressed in PL/SQL or using 3GL call outs). Attribute type can be: scalar (number, char, etc.), an ADT reference or a collection (nested table or array) [9].ADTs can be used as a column type or a table type. They can be used with relational operators. Constructors can be used to create instances of an ADT and definition would be done with CREATE TYPE. [8]

CREATE TYPE.
Creation of Object type is a simple 2-step process.
First, create the Object Type based on the columns of the relational table;

```
CREATE  TYPE  address_typ  as  OBJECT
 ( street      varchar2(120),
   city        varchar2(30),
   state       char(2),
   zip          number(5));
```

Second, create the table based on the Type and selecting rows from the Relational Table;

```
 CREATE  TABLE  customer1
 (customer_id     varchar2(5),
   last_name       varchar2(20),
   first_name      varchar2(14),
   address         address_typ);
INSERT INTO customer1
              ( customer_id ,     last_name  ,     "ADDRESS")
  VALUES (    '11111'     ,     'Smith'        ,      ADDRESS_TYP
                                          ('111 S. Main', 'Moscow', 'ID', 83843));
```

2.4 Collection Types

Collections are SQL data types that contain multiple elements. Each element or value for a collection is the same data type. In Oracle9i there are two collection types VARRAYs and nested tables. A VARRAY contains a variable number of ordered elements. VARRAY data types can be used as a column of a table or as an attribute of an object type. Also, the element type of a VARRAY may be either a native data type such as NUMBER or an object type. Using Oracle9i SQL, a named table type can be created. These can be used as nested tables to provide the semantics of an unordered collection. As with VARRAY, a nested table type can be used as a column of a table or as an attribute of an object type [2] [10].

2.4.1 Collection Types – Varrays

A VARRAY is an ordered collection of data. The number of data elements in the collection is pre-defined by the user. A VARRAY may be defined as a column in a database table, in which case the VARRAY data are stored in line with the 'main table'. Of particular significance, VARRAY data held in this manner cannot be indexed. Data within the VARRAY are manipulated as a whole [1] [4].

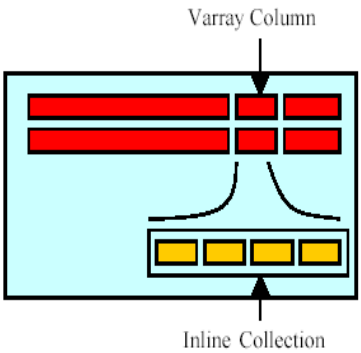
2.4.1.1 Creating Varrays table collections

Creating a Varray collection included in a database table is achieved in 2 steps. First, create an object type defining the 'layout' of the varray. (In this case it is the same as the child relational table except that the Foreign Key column - Order_ID has been removed.)

```
CREATE TYPE address_t AS VARRAY(10) OF VARCHAR2(30);
```

Second, create the database table, with the integrated varray as a field in the table .At this stage, the 'varray' collection could be manipulated locally within PL/SQL applications. As shown in figure(3).

```
create table   new_agent5(
        agent_id       varchar2(5 ),
        last_name      varchar2(25),
       first_name     varchar2(20),
       actor           address_t );
```



figure(3) varray

```
INSER INTO new_agent5
          (agent_id    , last_name    , first_name    , actor )
  values('11111'       ,'James'    , 'Hartman'     , address_t
                              ('111 S. Main', 'Moscow 222',));
```

2.4.2 Collection Types - Nested Tables

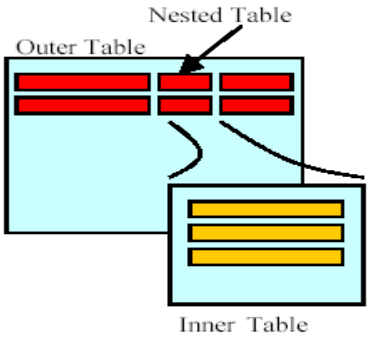
Nested tables are data collections of unbounded size ,allowing for limitless growth. When a table column is defined as a nested table, then logically we have an inner table resident within an outer table. Physically the inner table is a separate out-of-line table having the same general characteristics of relational tables. It is stored in the same table space as the outer table, taking by default the storage parameters of the table space. Data are not ordered in any particular sequence, and indexes may be applied[4]. Relationships between the inner and outer tables are automatically maintained within the database by using pointers. Given the unlimited size of the collection, there is more potential forth using Nested Tables, both inside the database and locally in PL/SQL [13]. Also, in the database there is the advantage of indexing the data and holding the data out of line. Nested tables may be useful for implementing master-detail type relationships [2].

2.4.2.1 Creating Nested table collections

Creating a Nested table collection included in a database table is achieved in 3 steps.

First, create an object type defining the 'layout' of the nested table. In this case it is the same as the child relational table except that the Foreign Key column - Order_ID has been removed.

```
CREATE TYPE address AS OBJECT
(address_type        char(1),
 street              varchar2(20),
 city                varchar2(20),
 state               char(2),
 zip                 varchar2(9));
```



figure(4) nested table

Second, create another type defined as a table
of the object type defined in the first statement.

```
CREATE TYPE  address_table  AS TABLE OF address;
```

At this stage, the 'nested table' collection could be manipulated locally within PL/SQL applications. The final step is to create the database table, the outer table, with the integrated nested table, and the inner table as shown in figure(4).

```
CREATE TABLE new_agent(
   agent_id          varchar2(5) not null,
   first_name        varchar2(20),
   last_name         varchar2(25),
   agent_address   address_table)
NESTED TABLE agent_address STORE AS address_tab
```

Retrieval and manipulation of data in the inner table is always achieved through the outer table. The following SQL statements, termed 'flattened sub-queries', show how to make use of the new SQL expression, to provide a link between the outer and inner tables;

```
insert into new_agent
        ( agent_id ,  first_name , last_name  , address_table)
values (  '11111'   , 'James'       , 'Hartman'   , address_table
      (address( 'H' , '1190 S.  Bend'  , 'Pullman' , 'WA' , '98119'),
       address( 'W' , '500 Main Street N.', 'Moscow' ,'ID'   , '83843')));
```

Here, agent_address is the column name and address_tab is the name of the physical table holding the nested table collection. The inner table is resident in the same table space as the outer table, and has the storage characteristics of the table space. Note that it is also possible to add a nested table column to an existing table similar to adding a new column to a relational table.

At this point, any queries against the inner table will access the data using a full table scan. So, despite the tight integration of the inner table with the outer table, it is still a table and so requires definition of proper access Nested Table Outer Table Inner Table paths [1] [4].

3. Analyzer

Oracle SQL analyzer provides the information about the database structure and modifies some initialization parameters to test the SQL statements against condition and data base environment.

As a SQL tuner, you need to be able to gather and analyze environmental data and performance statistics to help identify problem areas [7].

The following sections describe the information you can gather and the

methods Oracle SQL Analyze makes available for tuning the following statements.

Relation

Select Relation

```
SELECT c.street
   FROM   c.customer2 c WHERE c.customer_id=’22222’
```

Update Relation

```
UPDATE customer2 c
   SET c.zip=’22222’ WHERE c.customer_id=’22222’
```

Delete relation

```
DELETE   FROM customer2 c  WHERE   c.customer_id=’22222’
```

Correlated sub query

Select sub query

```
SELECT c.street  FROM addressr c
      WHERE  c.agent_id=(SELECT a.agent_id  FROM
new_agentr a
                          WHERE  a.agent_id=’22222’)
```

Update sub query

```
UPDATE addressr SET addressr.zip=’11987’ WHERE
addressr.zip=’22222’
 AND addressr.agent_id =(SELECT a.agent_id FROM new_agent a
                          WHERE  a.agent_id=’22222’)
```

Delete sub query

```
DELETE FROM addressr  WHERE  addressr.zip=’22222’
AND  addressr.agent_id= (SELECT  a.agent_id   FROM
new_agrntr a
                          WHERE a.agent_id=’22222’)
```

Object type (ADT)

Select ADT

```
SELECT c.”ADDRESS”.STREET ROM customer1 c WHERE
c.customer_id=’22222’
```

Update ADT

```
UPDATE customer1 c  SET c.customer_id=’11111’ WHERE
```

”ADDRESS”.ZIP=11111

Delete ADT

```
DELETE FROM customer1 c WHERE  c.”ADDRESS”.ZIP=11111
```

VARRAY

Update VARRAY

```
UPDATE new_agent5  SET actor = address_t (‘Engle 1111’,’Wood 222’)
    WHERE agent_id=’22222’
```

Delete VARRAY

```
DELETE FROM new_agent5 WHERE  agent_id=’22222’
```

Nested table

Update Nested table

```
UPDATE new_agent  SET agent_address = ADDRESS_TABLE(
    ADDRESS('H', '6042 Highland', 'Englewood','CO', '80112'),
    ADDRESS('W', '500 E. Arapahoe', 'Englewood', 'CO','80111'))
    WHERE agent_id = '22222'
```

Delete Nested table

```
DELETE  FROM  new_agent  WHERE agent_id = '22222'
```

3.1 Statistics & Discution

The task of the Oracle optimizer is to find the most efficient method for executing a SQL statement, but how do you know which optimizer mode will be most efficient for you statement ? Oracle SQL Analyzer lets you test each of these execution strategies against a SQL statement and provides performance statistics data that help you to determine which mode is best [7] .

Table(1) : Means of Measured Performance Parameter with the data types

Kind	Cost	Elapsed Time	CPU Time	Logical Blocks Read	Physical Blocks Read	UGA Memory	PGA Memory	Table Space
Nested Table	A 4855	A 1265	A 365037	A 26347936	A 13421		C B 284586	C 1.9
Varray	B 317	B 2	B 1	B 24139	B 3463	C 63830	A 530689	A 99657
Relation	C 196	E 1	B 1	D 12532	C 582	D C 55065	B 387502	B 93600
Object Type	D 93	D 1	B 1	E 11496	D 485	B 191440	C B 335865	C 29
Sub queries	E 62	C 1	B 1	C 13828	E 243	D 47515	C 260641	B 93601

***Numbers of same letter mean no significant difference**

Table(1) shows the performances of the data type structure processing parameters .Generally noticed that nested table is the mostly consuming time in case of elapsed time , cpu time , physical and logical read. But it still uses highest area of the memory as found at UGA (USER GLOBAL AREA) and PGA (PROGRAMING GLOBAL AREA) .But in reverse it consumes less table space for the other type, Except in the nested table , we found that varray , relational , type , sub queries have different letter from each other although they are statically different from each other (they have different Duncan's letter) at the parameter from cost until logical read . For table space parameter nested table take the same of type as they have the same Duncan's letter.

Table(2): Mean of measured performance with DML instructions

PROC	Cost	Elapsed Time	CPU Time	Logical Blocks Read	Physical Blocks Read	UGA Memory	PGA Memory	Table Space
Update	A 771.3	A 158.3	A 51211	A 3204880	A 2892	A 244198	B 409643	B 84240.5
Delet	B 312.5	B 46.9	B 7088	B 1094594	B 1607	B 127478	C 193795	C 77540.5
Select	C 150.6	C 0.3	B 0	C 1210	C 470	C 76015	A 552443	A 93600.5

***Numbers of the same letters mean no significant difference**

From table(2) notice that from all DML instructions the UPDATE one has the most effective in consuming time of operations and less effective in table space consuming . SELECT instructions have exactly the reverse effect and have dramatically less effective while DELETE instruction in

spite it differs from the other instructions, it nearly has the same effete of UPDATE.

Table(3):Represent the mean of measured performance with the size of processed data

Size	Cost	Elapsed Time	CPU Time	Logical Blocks Read	Physical Blocks Read	UGA Memory	PGA Memory	Table Space
1%	E 177	D 10.7	C 2743	D 591767	F 988	D 119034	B 292800	D 76731.3
5%	C 307	C 43.1	C 811	C 791194	C 1521	C 146938	A 420211	C 84404.8
10%	B 663	B 134.7	B 27108	B 2394515	A 2938	A 275689	AB 374954	E 75965.2
20%	A 1221	A 278.3	A 106321	A 5934814	B 2846	B 250893	A 408790	B 87651.6
40%	F 157	F 1.6	C 1	F 22453	E 1258	E 81849	AB 382173	B 87655.3
80%	D 181	E 3.4	C 2	E 41276	D 1294	F 65801	AB 314901	A 91160

*** Numbers of the same letters means no significant difference**

Data records size were measured by using 1,2,…..80 percentage and that as of 60,000 records , the mean of measured performance parameter from the cost to logical read was increased from 1% to 20% begin and begin to drop with 40% till 80% with a significant difference between them and that it may be caused by using different mechanism with size more than 12000 records .But looking to the consumed table space it had been found that it increased according to the increasing of data size and there is a significant difference between them.

Table(4) : Means of Measured Performance Parameter with the interaction between DATA Type & DML instruction

Kind	Cost	Elapsed Time	CPU Time	Logical Blocks Read	Physical Blocks Read	UGA Memory	PGA Memory	Table Space
Nested table& Update	A 6968	A 1801	A 588914	A 36119433	B 19021	A 2185555	B 455554	
Nested table & Delete	B 2319	B 621	B 96383	B 14622139	D 6702	B 563891	G 79425	
Nested table & Select								
Varray & Update	K 26	C 317	C 3	M 108	A 23225	M 0	G 69919	A 6775880
Varray & Delete	C 317	F 2	G 1	C 25053	E 3111	G 57741	C 361638	B 105715
Varray& Select								
Object Type & Update	G 188	G 2	F 1	H 16196	I 409	E 85291	D E 285125	

Object Type & Delete	**L 23**	**D 45**	**D 1**	**L 137**	**C 17557**	**D 86694**	**D C 343977**	
Object Type & Select	**I 45**	**L 0**	**M 0**	**J 1259**	**H 479**	**B 145052**	**A 708565**	
Relation & Update	**F 196**	**J 1**	**I 1**	**G 17780**	**J 368**	**H 47667**	**E 255737**	**F 93600**
Relation& Delete	**E 196**	**I 1**	**J 1**	**F 18496**	**G 537**	**I 47648**	**F 169039**	**H 93600**
Relation & Select	**D 196**	**K 0**	**K 0**	**I 1320**	**F 841**	**F 69881**	**A 737731**	**G 93600**
Sub queries & Update	**J 27**	**E 2**	**E 1**	**E 19310**	**K 358**	**L 47344**	**D CE 320407**	**E 93601**
Sub queries & Select	**M 1**	**H 1**	**H 1**	**D 21099**	**L 277**	**K 47572**	**F 172423**	**C 93601**
Sub queries & Delete	**H 158**	**M 0**	**L 0**	**K 1074**	**M 95**	**J 47629**	**DCE 289094**	**D 93601**

***Numbers of the same letters mean no significant difference**

Table (4) shows that the interaction between data type collect and DML order is noticed. Updating and deleting nested table have the most consuming elapsed time with the most of the measured parameter of performance , but the other data type collection consumes very less time in hundred times with any kind of DML order , but table space gives little difference with any of interaction combination.

Table(5) : Means of Measured Performance Parameter with the interaction between DATA size & DML instruction

Kind	Cost	Elapsed Time	CPU Time	Logical Blocks Read	Physical Blocks Read	UGA Memory	PGA Memory
Nested table 1	**D 329**	**J 88**	**C 23315**	**D 5001349**	**N 1874**	**D 356692**	**M 20741**
Nested table 5	**C 2245**	**C 617**	**D 11751**	**C 11403483**	**H 9207**	**C 1200766**	**A 774417**
Nested table 10	**B 4457**	**B 1140**	**B 230415**	**B 20295908**	**D 18353**	**B 1713398**	**H K G J L I 224733**
Nested table 20	**A 29770**	**A 7768**	**A 2976980**	**A 165831301**	**A 48310**	**A 5250888**	**B 599720**
Varray 1	**P 159**	**F 159**	**P 1**	**O 9004**	**L 4609**	**Y 30580**	**K L 163097**
Varray 5	**O 161**	**I 159**	**R 1**	**W 4021**	**J 6258**	**X 33589**	**H D F G J E I 292209**
Varray 10	**N A64**	**H 159**	**O 1**	**R 5761**	**I 7950**	**B 23548**	**H K F G J L I 253574**
Varray 20	**M 169**	**G 159**	**K 1**	**L 11250**	**G 11058**	**A 23548**	**H K F G J L I 253574**
Varray 40	**L 179**	**E 159**	**G 2**	**K 16209**	**E 17866**	**Z 23648**	**K J L 190846**
Varray 80	**E 199**	**D 160**	**E 6**	**G 29239**	**B 31268**	**W 38308**	**L 141368**
Object Type 1	**W 78**	**P 15**	**B 0**	**B 832**	**P 871**	**E 184272**	**B 655160**
Object Type 5	**V 79**	**N 15**	**V 0**	**Y 2916**	**O 1851**	**H 83157**	**D F C E 358791**
Object Type 10	**U 81**	**O 15**	**T 0**	**V 4058**	**M 2861**	**J 81845**	**D C E 388323**
Object Type 20	**T 84**	**M 16**	**N 1**	**T 5506**	**K 5004**	**I 82296**	**H D F G E 332162**
Object Type 40	**S 91**	**L 16**	**I 2**	**Q 6763**	**F 11318**	**F 84503**	**C 472582**
Object Type 80	**R 104**	**K 17**	**F 2**	**J 17903**	**C 18380**	**G 84420**	**H D F G J E I 289916**

Relation 1	**G 196**	**Z 0**	**A 0**	**A 1723**	**R 679**	**P 54795**	**D F G E 344016**
Relation 5	**I 196**	**A 0**	**Y 0**	**X 3448**	**V 362**	**L 54833**	**D C 398940**
Relation 10	**J 196**	**Y 0**	**W 0**	**S 5604**	**U 394**	**M 54833**	**D C 395604**
Relation 20	**F 196**	**W 1**	**S 0**	**N 9916**	**T 614**	**N 54833**	**D C 398619**
Relation 40	**H 196**	**T 1**	**M 1**	**I 18657**	**Q 780**	**K 56268**	**D C E 389213**
Relation 80	**K 196**	**R 2**	**J 1**	**F 35844**	**S 664**	**O 54831**	**D C 398619**
sub queries 1	**Y 53**	**B 0**	**Z 0**	**Z 1975**	**A 178**	**T 47486**	**K J L I 208345**
sub queries 5	**X 53**	**U 1**	**X 0**	**U 4655**	**Y 289**	**Q 47570**	**H K J L I 217207**
sub queries10	**Z 53**	**X 1**	**U 0**	**P 7515**	**B 105**	**S 47528**	**H D F G J E I 303094**
sub queries20	**A 53**	**V 1**	**Q 1**	**M 10965**	**W 334**	**U 47479**	**H K F G J E I 272938**
sub queries40	**B 53**	**S 2**	**L 1**	**H 21099**	**X 313**	**V 47465**	**H K F G J L I 239792**
sub queries80	**Q 106**	**Q 3**	**H 2**	**E 36759**	**Z 240**	**R 47562**	**H D F G E I 322471**

- **Numbers of the same letters mean no significant difference**

Table (5) gives the of mean values of measured parameters data type collection with data size interactions. We can see in table(1) that nested table is the most effective in consuming processing time with every size of data ,this effect increased as the size increased. Nested table with the size of 40% and 80% their measured parameter could not be done because of huge time of execution. And this is true from interaction of any of data type collection with the data size for most of the measured parameters.

Conclusion:

From this paper we conclude that:

- SQL analyzer is a good tool for studying optimum performance of SQL statements using statistics.
- Using statistical analyze represented by ANOVA table were very helpful in judgment between compromises , it enables to group the effects of data type , data size & DML language individually . Without any interference between each other , also we can get the interaction rule between data type and data size and DML. We found that nested table was the most effective from other types of data type , the object type was the least effective (consume lowest time of execution). We also found that increasing data size will increase any of performance data parameter until we reach size of 40% the time begin to degrees.
- Update statements show the most effective in consuming execution time & table space then Delete & Select statements, respectively.

REFERENCES

[1] دايتر كارول , 2000 , اوراكل 8 باييل , دار الفاروق للنشر والتوزيع / مركز التعريب والترجمة / مصر.

[2] Johnny O. , Allan R.L ,2000 , Experiences from object_Relational Programing Oraclre8 COT/4 - 4/V1.0 , http://www.citdk/COT/reports/case4/.4v1.COT-4-04-1.pdf

[3] Cornwell I. & Coleman J. , 1999,Review of Data Base Technologies UTMC-10 Report 1,12.

[4] Lonsdale M. , 1999 , Is Performance a Reson for using Oracle8 Object http://www.softlab.co.uk/news/uploads/ISPERF08.pdf.

[5] Nori A. ,Bringing Object_Relational Technology to the Main Stream Vertis Corporation , University Avenue , Palo Alto , CA.

[6] Oracle_Developer , 2000, Introduction to Object _Relational Data Base Development, http://www.Kingtraining.com/downloads/08 diffs paper.pdf

[7] Oracle Enterprise Manager Data Base , 2001 , Tuning with the Oracle tuning pack , http://www.cs.umb.edu/cs634/oracl9i/em.920/986647/vmqintro.htm.

[8] Oracle Technical White Paper , 2001 , Simple Strategies for complex data : Oracle9I Object Relational Technology , http://www.ont.oracle.com/products/oracle9i/pdf/idt_twp.pdf

[9] Russell J. ,1999, Application Developers Guide Object-Relational Feature , http://www.thinkspark.co.uk/ioag/plsqlnews .pdf

[10] Sikora Y. , Peter J. ,2001 , Object to Object Communication , http://www.odtug.com.

[11] SAS Institute Inc.,SAS Camus Drive .Cary,North Carolina USA

[12] Weber M. , SQL tuning for Oracle , mweber11@earthlink.net.

[13] Whithead A. ,2000, Object Relational Oracle8 and PL/SQL8 , http://www.polito/itivrea/informazioni/passaggio/dispenseinseif/basic/plsql8.pdf.