بسم الله الرحمن الرحيم

ربط الاوراكل بالفيجوال بيسك

ORACLE AND VISUAL BASIC

تأليف
محمد ابراهيم العضيب
١٤٢٣هـ/٢٠٠٣ م
www.alabrags.com

أهداء

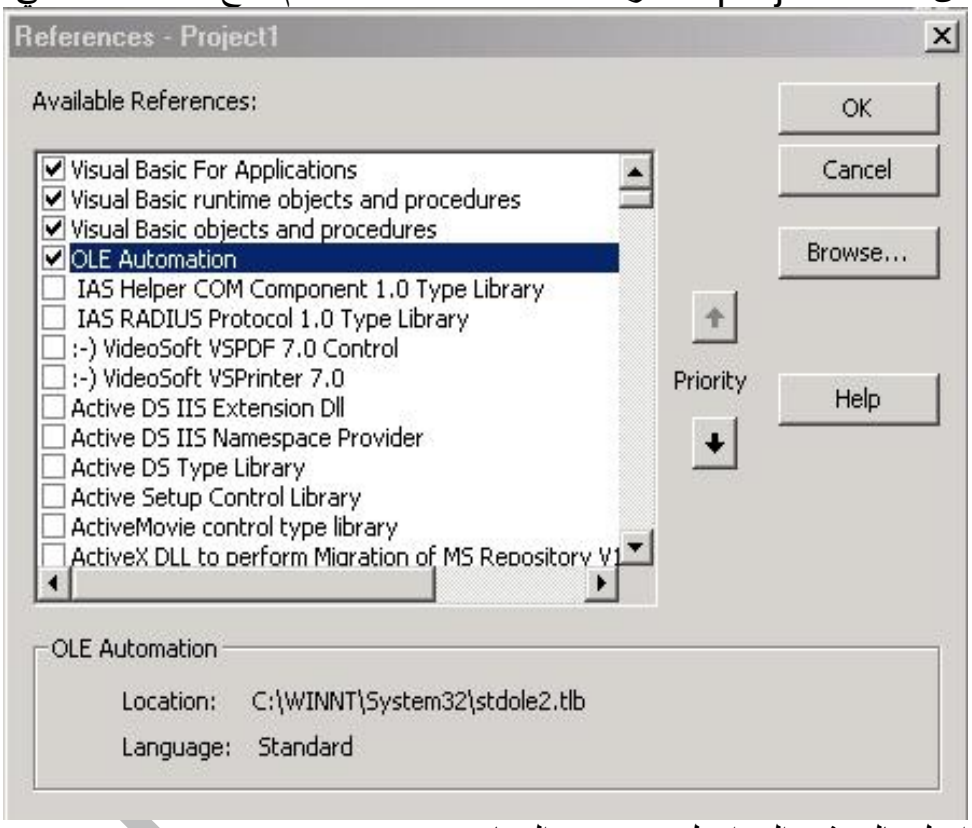
إلى كل ابناء الشعب العربي الكريم اهدي لكم هذا الكتاب الفريد والذي يقوم بانشاء واجهات لقواعد البيانات المصممة بالاوراكل واستخدام الاداة OO4O التي توفرها الاوراكل
وكما انا للمال زكاة فان للعلم زكاة ايضا لكن زكاة العلم ليس بالمال ولكن بتعليم هذا العلم
واتمنى من كل شخص يتقن اي علم وفي اي مجال ان يقوم بتعليم هذا العلم
لأن ذلك سيعود على امتنا بالنفع والرقي وكلنا نعرف انا اغلب المبتكرات تم معرفتها من قبل علماء عرب مثل الحسن بن هيثم، الخوارزمي ، ابن حيان وغيرهم.
وننشر هذا العلم الى جميع ابناء العرب دون قيود او شروط ولا تكن المادة هي مصلحتنا من ذلك بل نجعل هدفنا تطوير ابناء العرب وجعلهم في سباق مع الزمن .
واتمنى من للجميع الاستفادة
ولاتنسونا بخالص دعائكم

تمهيد

نستخدم الاداة OO40 لربط الاوراكل وتوجد هذه الاداة ضمن مجلدات الاوراكل
وتعبر هذه الاداه من اقوى الادوات على ربط الاوراكل بالفيجوال بيسك.
بعد الانتهاء من هذا الكتاب يكون باستطاعتك
انشاء تطبيقات وواجهات لقواعد البيانات
وسوف نتطرق لأغلب الادوات الموجودة في هذه الاداة.

الدرس الاول: اضافة الاداة OO40 الى المشروع.

يلزم اضافه هذه المكتبة الى المشروع قبل البدء بكتابه اي كود
ولكي نضيف هذه المتكبة نقوم بمايلي:
من قائمة project نختار REFERENCES ثم تفتح لنا قائمة كالتالي:



انزل بالموشر الى اسفل حتى تجد الخيار
Oracle objects for OLE 3.0 type library
ضع الموشر علية ثم قم باختياره
واذا لم تجد هذا الاختيار قم بمايلي
اذا لم تجدها هنالك فصلح browser من قائمة reference السابقة ثم اذهب الى المجلد اللي قمت بتنزيل الاوراكل عليه ثم ادخل على مجلد bin ثم ابحث في الملفات عن ملف اسمة oip8.tlb اختره وسوف تتضاف لك الاداة.

الدرس الثاني: الاشياء التي تتضمنها هذه الاداة

١- orasession:

هذا الشيء يمثل الاساس في النموذج OO40 وهو يقوم بتشغيل التطبيق وانشاء نسخة من الخادم INPROC SERVER وهو يحتوي على وسائل لانشاء الاتصال بقاعدة بيانات اوراكل.

٢- oradatabase :

يمثل اتصالا بقاعدة بيانات اوراكل وهو يقدم وسائل لتنفيذ عبارات sql

---

٣- orasqlstmt :

وهو يستخدم لتنفيذ عبارات DDL مثل CREATE TABLE

٤- oradynaset :

يمثل النتائج المعادة من عبارة sql وهو مؤشر في جانب العميل يمكن معالجته والتعامل معه
ويتم انشائه باستخدام العبارة createdynaset للشيء oradatabase
وهنالك المزيد من الاشياء لكن سوف نتطرق لك واحدة في حينها

٥- oraparameters :

يمثل متغير ربط في عبارة sql وهو يقدم وسائل لإنشاء ومعالجة قيم المعاملات
(يستخدم لتمرير متغيرات الاجراءات والدوال وعبارات sql)

٦- oraobject :

يمثل نسخة من اشياء اوراكل المخزنه على شكل جدول

٧- oraparametersarray

مثل الشيء oraparameters لكن يكون مصفوفة من المعاملات والمتغيرات

الدرس الثالث/ بدء العمل مع الاداة وطريقة الاتصال

طريقة الاتصال بقاعدة بيانات اوراكل:

اولا:

نضيف قبل كل شي الاداة من قائمة project ثم reference ثم ابحث عن oracle objects for OLE 3.0 type library

ثم نقوم بتعريف ثلاث متغيرات :

الاول متغير من نوع oraseesion وهو لتكوين سيرفر مع القاعدة وفتح القاعدة.

الثاني متغير من نوع oradatabase وهو للاتصال مع قاعدة البيانات.

الثالث متغير من نوع oradynaset لتخزين السجلات المستنتجة.

ويمن التعريف كمايلي:

```
Dim ses As orasession
Dim database As oradatabase
Dim dyn As OraDynaset
```

ثانيا/

نكون inprocserver والاتتصال بقاعدة البيانات وهذه ثابته دائم لاتتغيرفي أي عملية اتصال لابد من وجودهما

```
Set ses = CreateObject("oracleinprocserver.xorasession")
```

ثم نقوم بفتح قاعدة البيانات كمايلي:

```
Set database = ses.OpenDatabase("ddd"," scott/tiger",0&)
```

حيث:

Ses : هذا متغير orasession الذي تم فتحه سابقا.

OpenDatabase : وسيلة لفتح قاعدة البيانات.

ddd : هذا اسم قاعدة البيانات لديك وهو الاسم الذي اعطيته لقاعدة البيانات اثناء تنزيل الاوراكل.

" scott/tiger" : هذا اسم الحساب وكلمة المرور وهو افتراضي في اي قاعدة بيانات يوجد بها هذا المستخدم وكلمة مرورة .او يكون لك اسم مستخدم خاص بك وكلمة مرور له.

0& تمثل حالة القاعدة الابتدائية.

والعمليتان السابقتين وهما فتح الاتصال وفنح القاعدة لابد منهما في اي كود لكي تتصل بقاعدة البيانات.

ثالثا/

نقوم الان بكتابة الاستعلام الذي نريد من أي جدول في قاعدة البيانات كمايلي

```
mysql="select * from emp"
```

رابعا/

كيفيه تشغيل الاستعلام(فتح الاستعلام) وتخزين النتائج كما يلي

```
Set dyn = database. CreateDynaset (mysql, 0&)
```

وذلك باستخدام الامر CreateDynaset للشي oradatabase

حيث :

Mysql : تمثل المتغير الذي تم تخزين الاستعلام به وليس شرطا ان يكون نفس الاسم .

خامس/

بعد تنفيذ الاستعلام ولو فرضنا ان الجدول في الاستعلام السابق يحتوي على حقلين الاول empno والثاني ename ونريد بعد تنفيذ الاستعلام ان يقوم بعرضهما في مربعين نص على النموذج ويكون كمايلي:

```
Text1.Text = dyn.Fields("empno").Value
Text2.Text = dyn.Fields("ename").Value
```

وبالتالي يكون الكود كامل كمايلي:

```
Dim ses As orasession
Dim database As oradatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
sql = "select * from emp"
Set dyn = database.CreateDynaset(sql, 0&)
Text1.Text = dyn.Fields("empno")
Text2.Text = dyn.Fields("ename")
```

وبهذا انهينا طريقة الاتصال بالقاعدة والان لنبدء الابحار في وسائل هذه الاداة

‘‘

الدرس الرابع/ اضافة ، تحديث ، حذف البيانات.

لنقم بتكوين جدول للقيم بالتطبيق عليه ولنفرض ان الجدول كمايلي:

S_no	S_name	job	salary
الرقم	الاسم	الوظيفة	الراتب

ولانشاء هذا الجدول نذهب الى sql * plus ونكتب الامر

```
Create table employee(
S_no char(9),
S_name varchar2(50),
Job varchar2(50),
Salary number(10));
```

وبهذا نكون قد انشاءنا الجدول

أ- الاضافة ADDNEW :

هذه الوسيلة تسمح باضافة الصفوف الجديدة إلى dynaset الذي قمنا بانشاءه ولكي نضيف صف جديد الى الجدول employee السابق نقوم بمايلي.

اولا نقوم بفتح الاتصال وتكوين dynaset كمايلي:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from employee"
Set dyn = database.CreateDynaset(SQL, 0&)
```

ولكي تضيف صف نقوم بفتح dynaset للاضافة بالوسيلة addnew كمايلي:

```
dyn.AddNew
```

ثم نقوم باسناد القيم الجديدة لحقول dynasetكمايلي:

```
dyn.Fields("s_no").Value = "1000"
dyn.Fields("s_name").Value = "ddd"
dyn.Fields("job").Value = "programming"
dyn.Fields("salary").Value = "5500"
```

ثم نقوم بتثبيت هذه المدخلات بالقاعدة باستخدام الامر update لـ dynaset كمايلي:

```
dyn.Update
```

وبالتالي يصبح الكود كامل كمايلي.

الكود كامل :

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from employee"
Set dyn = database.CreateDynaset(SQL, 0&)
dyn.AddNew
dyn.Fields("s_no").Value = "1000"
dyn.Fields("s_name").Value = "MOHAMMED"
dyn.Fields("job").Value = "programing"
dyn.Fields("salary").Value = "5500"
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

ويمكنك تطوير البرنامج السابق وجعله اكثر شمولية وذلك بجعل المدخلات يقوم بادخالها المستخدم وذلك كمايلي :

بانشاء فورم جديد ووضع به اربع مربعات نص وزر امر وداخل زر الامر اكتب الكود التالي:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from employee"
Set dyn = database.CreateDynaset(SQL, 0&)
dyn.AddNew
dyn.Fields("s_no").Value = TEXT1.TEXT
dyn.Fields("s_name").Value = TEXT2.TEXT
dyn.Fields("job").Value = TEXT3.TEXT
dyn.Fields("salary").Value = TEXT4.TEXT
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

وبالتالي فان المكتوب باللون الاحمر هو الذي تم تبديله عن الكود السابق

قم بتطبيق المثال طبقا للخطوات السابقة وسوف تسعد بالنتائج....

ب – التعديل UPDAET :

لتعديل اي سجل موجود بالقاعدة نستخدم الامر edit لـ dynaset
كمايلي:
اولا نقوم بفتح الاتصال وتكوين dynaset كمايلي :

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from employee where s_no=1000"
Set dyn = database.CreateDynaset(SQL, 0&)
```

ثم نقوم بعملية التعديل كمايلي :
نقوم اولا بجعل سماحية التعديل على dynaset كمايلي

```
Dyn.edit
```

ثم نقوم باجراء التغير كمايلي

```
Dyn.fields("s_name").value="ali"
```

ثم نقوم بثبيت التعديل باستخدام الامر update لـ dynaset كمايلي:

```
Dyn.update
```

لناخذ مثال على ذلك :
في جدول employee قم بتعديل راتب الشخص الذي رقمه ١٠٠٠ ؟
ويكون كود التعديل كمايلي:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from employee where s_no=1000"
Set dyn = database.CreateDynaset(SQL, 0&)
Dyn.edit
Dyn.fields("salary").value="6000"
Dyn.update
MsgBox "تم التعديل"
```

ويهذا يكون تم التعديل على قاعدة البيانات.

ج- الحذف DELETE :

لحذف اي سجل موجود في القاعدة نستخدم delete لـ dynaset
وللبداية بحذف سجل
اولا نقوم بفتح الاتصال وتكوين dynaset كمايلي:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from employee where s_no=1000"
Set dyn = database.CreateDynaset(SQL, 0&)
```

ثم نقوم بعملية الحذف كمايلي:

```
Dyn.delete
```

وبالتالي يصبح الكود كامل كمايلي:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from employee where s_no=1000"
Set dyn = database.CreateDynaset(SQL, 0&)
dyn.Delete
MsgBox "تم الحذف"
```

وبهذا يتم الحذف

الدرس الخامس/التنقل بين السجلات

** الوسائل MOVE

أ - الوسيلة MOVEFIRST :
تنقل مؤشر الصفوف من موضعه الحالي إلى بداية مجموعة السجلات

ب - الوسيلة MOVELAST :
تنقل مؤشر الصفوف من موضعه الحالي إلى نهاية مجموعة السجلات

ج- الوسيلة MOVENEXT :
تنقل مؤشر الصفوف من موضعه الحالي إلى الصف التالي في مجموعة السجلات.

د- الوسيلة MOVEPREVIOUS :
تنقل مؤشر الصفوف من موضعه الحالي إلى الصف السابق في مجموعة السجلات.

هـ- الوسيلة MOVEPREVIOUSN :
هذه الوسيلة تنتقل مؤشر الصفوف من موقعة الحالي إلى الموضع قبل الصف الحالي بعدد n من الصفوف في مجموعة السجلات dynaset أي تنقل المؤشر إلى الخلف

مـــثـــــال:
افتح مشروع جديد في الفيجوال بيسك وقم بوضع صندوق نص وزر امر فقط
ثم اضغط ضغطه مزدوجة على زر الامر واكتب الكود التالي :

```
1 Dim Ses As OraSession
2 Dim Database As OraDatabase
3 Dim Dyn As OraDynaset
4 Set Ses = CreateObject("OracleInProcServer.XOrasession")
5 Set Database = Ses.OpenDatabase("ddd", "scott/tiger", 0&)
6 Set Dyn = Database.CreateDynaset("Select * from emp", 0&)
7 Dyn.MoveLast
8 Dyn.MovePreviousn (2)
9 Text1.Text = Dyn.Fields("ENAME").Value
```

الاسطر ١- ٦ فتح وتكوين الاتصال بالقاعدة ومن ثم تكوين مجموعة السجلات
السطر ٧ يقوم بالذهاب بمؤشر الماوس إلى أخر صف
السطر ٨ يقوم بتحريك المؤشر صفين إلى الخلف
السطر ٩ يضع في خانة النص الموجودة في النموذج اسم الموظف الذي المؤشر متوقف عنده

فلو كان جدول emp كمايلي:

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

فبعد تنفيذ الكود السابق فعتدما يصل الى السطر رقم ٧ في الكود سوف يكون التالي:

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

المؤشر سوف يكون هنا

وبعد تنفيذ الخطوة رقم ٨ سوف يقوم المؤشر بالرجوع إلى الخلف صفين كمايلي

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

المؤشر سوف يكون هنا

وبالتالي سوف يتم عرض اسم ALI في مربع النص

و- الوسيلة MOVENEXTN :

هذه الوسيلة تنتقل مؤشر الصفوف من موقعة الحالي إلى الموضع بعد الصف الحالي بعدد n من الصفوف في مجموعة السجلات dynaset أي تنقل المؤشر إلى الامام

مـثـــــال :

افتح مشروع جديد في الفيجوال بيسك وقم بوضع صندوق نص وزر امر فقط
ثم اضغط ضغطه مزدوجة على زر الامر واكتب الكود التالي :

```
1 Dim Ses As OraSession
2 Dim Database As OraDatabase
3 Dim Dyn As OraDynaset
4 Set Ses = CreateObject("OracleInProcServer.XOrasession")
5 Set Database = Ses.OpenDatabase("ddd", "scott/tiger", 0&)
6 Set Dyn = Database.CreateDynaset("Select * from emp", 0&)
7 Dyn.Movefirst
8 Dyn.Movenextn (2)
9 Text1.Text = Dyn.Fields("ENAME").Value
```

الاسطر ١- ٦ فتح وتكوين الاتصال بالقاعدة ومن ثم تكوين مجموعة السجلات
السطر ٧ يقوم بالذهاب بمؤشر الماوس إلى أول صف
السطر ٨ يقوم بتحريك المؤشر صفين إلى الخلف
السطر ٩ يضع في خانة النص الموجودة في النموذج اسم الموظف الذي المؤشر متوقف عنده فلو كان جدول emp كمايلي:

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

فبعد تنفيذ الكود السابق فعتدما يصل الى السطر رقم ٧ في الكود سوف يكون التالي:

المؤشر سوف يكون هنا

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

وبعد تنفيذ الخطوة رقم ٨ سوف يقوم المؤشر بالتقدم إلى الامام صفين كمايلي

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

المؤشر سوف يكون هنا

وبالتالي سوف يتم عرض اسم FAHAD في مربع النص

ي- الوسيلة MOVEREl:

هذه الوسيلة تنقل مؤشر الصفوف من موضعة الحالي الى الصف في الموضع على بعد n بالنسبة للصف الحالي.والفرق بين هذه الوسيلة والوسيلة MOVENEXTN ان الوسيلة MOVEREl تقبل الاعداد الموجبة والسالبة. فإذا كان العدد موجبا انتقل مؤشر الصفوف إلى الامام وإذا كان سالب انتقل مؤشر الصفوف الى الخلف

مـــثـــــال/

افتح مشروع جديد في الفيجوال بيسك وقم بوضع صندوق نص وزر امر فقط
ثم اضغط ضغطه مزدوجة على زر الامر واكتب الكود التالي :

```
1 Dim Ses As OraSession
2 Dim Database As OraDatabase
3 Dim Dyn As OraDynaset
4 Set Ses = CreateObject("OracleInProcServer.XOrasession")
5 Set Database = Ses.OpenDatabase("ddd", "scott/tiger", 0&)
6 Set Dyn = Database.CreateDynaset("Select * from emp", 0&)
7 Dyn.Movefirst
8 Dyn.Moverel (3)
9 Text1.Text = Dyn.Fields("ENAME").Value
```

الاسطر ١- ٦ فتح وتكوين الاتصال بالقاعدة ومن ثم تكوين مجموعة السجلات
السطر ٧ يقوم بالذهاب بمؤشر الماوس إلى أول صف
السطر ٨ يقوم بتحريك المؤشر ثلاث صفوف الى الامام ولو كان العدد 3- لكان المؤشر يتحرك ثلاث صفوف الى الخلف
السطر ٩ يضع في خانة النص الموجودة في النموذج اسم الموظف الذي المؤشر متوقف عنده
فلو كان جدول emp كمايلي:

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

فبعد تنفيذ الكود السابق فعندما يصل الى السطر رقم ٧ في الكود سوف يكون التالي:

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

المؤشر سوف يكون هنا

وبعد تنفيذ الخطوة رقم ٨ سوف يقوم المؤشر بالتقدم الى الامام ثلاثة صفوف كمايلي

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

المؤشر سوف يكون هنا

وبالتالي سوف يتم عرض اسم LUAI في مربع النص

• الوسائل FIND

هذه الوسائل تسمح لك بالبحث خلال مجموعة السجلات dynaset والانتقال إلى الصف الذي يطابق الشروط المحددة في العبارة find ويمكنك استخدام التعبيرات في العبارة find التالية :

الاستفسارات البسيطه مثل 'name='MOHAMMED<

الاستفسارات المحتوية على تعابير معقدة مثل:<

Salary+addcomm >5000

استدعاء وظائف sql مثل UPPER(NAME)='sami'

١- الوسيلة FINDFIRST :

تصل إلى اول صف في dynaset الذي يطابق العبارة find

فمثلا لو كان لدينا الجدول التالي ونرغب في الحصول على اسم اول موظف راتبه 6500

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000
1004	FADY	ANALYST	6500
1005	Adel	MANAGER	6500
1006	fuad	ANALYST	5350

وبالتالي فأن الكود سيكون كمايلي

الكود:

```
1 Dim ses As orasession
2 Dim database As oradatabase
3 Dim dyn As OraDynaset
4 sql = "select *from emp"
5 Set ses = CreateObject("oracleinprocserver.xorasession")
6 Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
7 Set dyn = database.CreateDynaset(sql, 0&)
8 sql_condition = "SAL=3000"
9 dyn.FindFirst sql_condition
10 text1.text= Dyn.Fields("ENAME").Value
```

الجديد في هذا الكود هو السطر رقم 9 وهو تحديد معيار البحث باسم الحقل والقيمة التي يبحث عنها.

وبالتالي سيكون الناتج هو الاسم Ali

٢- الوسيلة FINDLAST :

تصل إلى أخر صف في dynaset الذي يطابق العبارة find
وهذا عكس FINDFIRST .
فمثلا لو كان لدينا الجدول التالي ونرغب في الحصول على اسم أخر موظف راتبه 6500

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000
1004	FADY	ANALYST	6500
1005	Adel	MANAGER	6500
1006	fuad	ANALYST	5350

والكود كمايلي :

```
1 Dim ses As orasession
2 Dim database As oradatabase
3 Dim dyn As OraDynaset
4 sql = "select *from emp"
5 Set ses = CreateObject("oracleinprocserver.xorasession")
6 Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
7 Set dyn = database.CreateDynaset(sql, 0&)
8 sql_condition = "SAL=3000"
9 dyn.FindLAST sql_condition
10 text1.text= Dyn.Fields("ENAME").Value
```

الجديد في هذا الكود هو السطر رقم 9 وهو تحديد معيار البحث باسم الحقل والقيمة التي يبحث عنها.
وبالتالي سيكون الناتج هو الاسم **Adel**

٣- الوسيلة FINDNEXT :

تصل الى الصف التالي في dynaset

٤- الوسيلة FindPrevious :

تصل الى الصف السابق في dynaset

الدرس السادس/الوسيلة EXECUTESQL

تنفذ عبارات sql او pl/sql وهذه الوسيلة هي لـ oradatabase

مثال لوكان لدينا الجدول التالي

Empno	Ename	Job	salary
1000	SAMI	ANALYST	4500
1001	ALI	MANAGER	6500
1002	FAHAD	MANAGER	6600
1003	LUAI	SALESMAN	7000

ونريد تعديل راتب الموظف الذي رقمه 1002 من 6600 الى 7500 وبالتالي يكون الكود كمايلي:

```
1 Dim ses As orasession
2 Dim database As oradatabase
3 Dim dyn As OraDynaset
4 Set ses = CreateObject("oracleinprocserver.xorasession")
5 Set database = ses.OpenDatabase("ddd"scott/tiger", 0&)
6 sql = "update emp set sal=7500 where EMPNO=1002"
7 database.ExecuteSQL sql
8 MsgBox"تم التعديل"
```

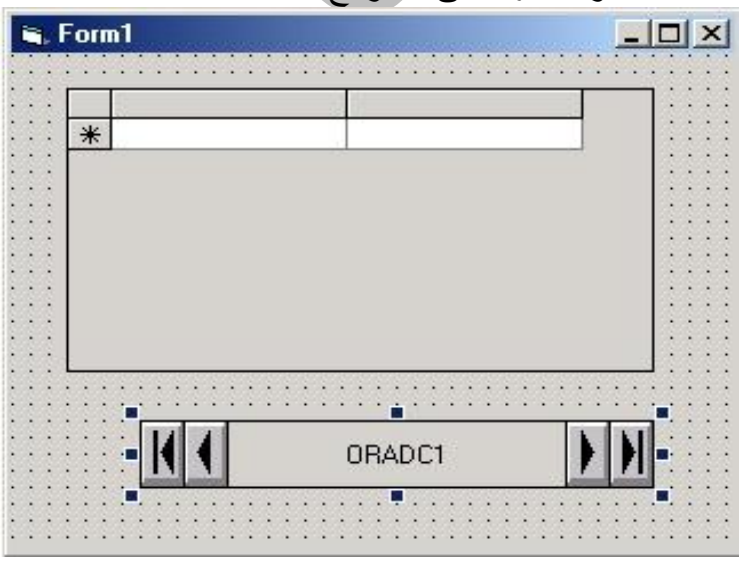
وبهذ يتم التعديل.

ويمكن ايضا استخدام هذه الوسيله في عبارات insert.

الدرس السابع/اداة التحكم فى اوراكل oracle data control

هذه الاداة تسمح لك بتنفيذ معظم عمليات التعامل مع البيانات دون كتابة سطر كود واحد.
ولتنفيذ استفسار بسيط باستخدام عمليات السجلات الديناميكة فانك تحتاج إلى بناء تطبيقي يحتوي على عدة سطور وتحتاج الى ازرار التحرك لكن مع استخدام هذه الاداة يكون الامر اكثر بساطة
ولعرض جدول emp باستخدام oracle data control والاداة Microsoft data Bound gride control نقوم بمايلي:

١- من قائمة project نختار components ثم نختار oracle data control
٢- من قائمة project نختار components ثم نختار Microsoft data Bound gride control
٣- ثم نقوم بالذهاب الى مربع الادوات ونختار الاداة Microsoft data Bound gride control ونسحبها على النموذج
٤- ثم نقوم بالذهاب الى مربع الادوات مرة اخرى ونختار الاداة oracle data control ونسحبها على النموذج



ومن ثم نذهب الى الخصائص ونضبط الخصائص كمايلي:

اولا/ خصائص oracle data control

* connect

هذه الخاصة تحدد اسم المستخدم وكلمة المرور مثل"scott/tiger "

* databasename

تحدد اسم قاعدة البيانات مثل "ddd" حسب ماقمت بتسميتها لديك

* recordsource
تحدد الاستعلام المطلوب مثل
"select * from emp"

ثانيا/ربط الادة Microsoft control مع oradc1 بضبط خاصية
Datasource للاداة Microsoft control على oradc1
ثم بعد ذلك نفذ البرنامج واستمتع بالنتائج
لاحظ سهولة التعديل والتنقل ايضا.

ألم يتبادر إلى ذهنك لو اردنا استبدال الاداة Microsoft control بمربعات نص كيف يتم العمل ؟

العمل من اسهل مايكون ويتم كمايلي:
نقوم بانشاء مشروع جديد ثم نقوم بادراج الاداة oracle data control (كماسبق تعلمنا كيفية ادراجها)
ثم ندرج اربعة مربعات نص الى سطح النموذج ثم نقوم بضبط الخواص كمايلي:
اولا/خواص oracle data control

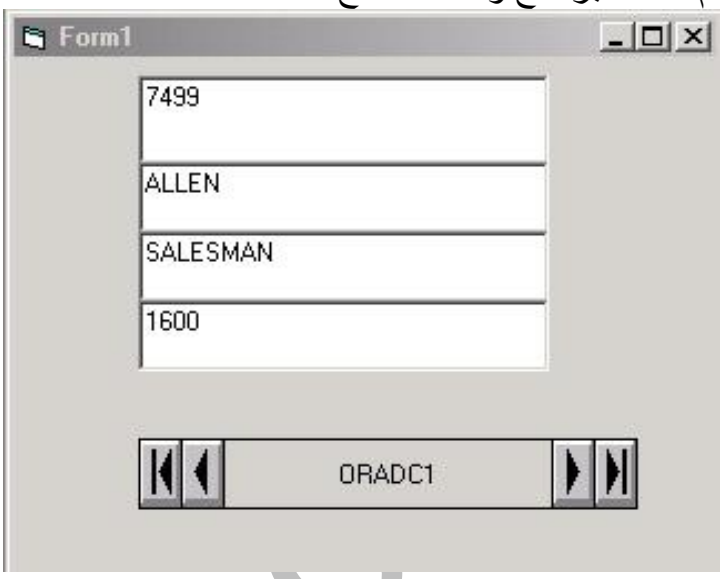
* connect
هذه الخاصة تحدد اسم المستخدم وكلمة المرور مثل"scott/tiger "

* databasename
تحدد اسم قاعدة البيانات مثل "ddd" حسب ماقمت بتسميتها لديك

* recordsource
تحدد الاستعلام المطلوب مثل
"select * from emp"

ثانيا/ خواص مربعات النص:
- مربع النص الاول :
خاصية datasource نضبطها الىoradc1
خاصية datafield نضبطها الى اسم الحقل مثلا empno
- مربع النص الثاني :
خاصية datasource نضبطها الىoradc1
خاصية datafield نضبطها الى اسم الحقل ename
- مربع النص الثالث :
خاصية datasource نضبطها الىoradc1
خاصية datafield نضبطها الى اسم الحقل job
- مربع النص الرابع :
خاصية datasource نضبطها الىoradc1

خاصية datafield نضبطها الى اسم الحقل sal
ثم شغل البرنامج وشاهد النتائج



الدرس الثامن/ التجميعات VARRAY ,NESTED TABLE

النموذج OO40 يوفر طريقة ممتازة للتعامل مع الجداول التي تشتمل على الاشياء OBJECT والتجميعات COLLECTION .

ويوفر النموذج OO40 وسيلة تسمى createoraobject للشي oradatabase وعند تنفيذ هذه الوسيلة فانها تعيد object جديد او collection جديد ونأخذ مثال على object لو اردنا انشاء جدول الموظفين والذي يحتوي على رقم الموظف وتاريخ الالتحاق بالوظيفة وانشاء object لتخزين مسمى الوظيفة والراتب وبالتالي يكون الجدول مكوّن من ثلاث حقول رقم الموظف ،اسمه ،(مسمى الوظيفة والراتب)

ولذلك يجب انشاء اولا النوع object ويكون كمايلي:

```
Create or replace type job as object
( jobname varchar2(20),
salary number(5));
/
```

وبهذا نكون قمنا بانشاء النوع الجديد وهو يحتوي على اسم الوظيفة والراتب. الان نقوم بانشاء الجدول:

```
Create table employee
(empno number,
empname varchar2(50),
jobemp  job);
```

لاحظ اخر حقل كيف تم تعريفه اسمه jobemp ونوعة job وهو النوع الذي قمنا بانشاءه سابقا ويمكننا اضافه للجدول

```
Insert into employee
values(414,'mohammed',job('manager',6000));
```

انواع التجميعات في قاعدة البيانات:

التجميعة هي مجموعة من العناصر من نفس النوع

والتجميعات هي على نوعين:

- **Ø** التجميعة varray

وهي كمصفوفة متغيرة ومشابهة للمصفوفات في اي لغة من لغات البرمجة مثل c و++c وتتم الاشارة الى اي عنصر في هذه التجميعة باستخدام الارقام السفلية ويتم التخزين في هذه التجميعة بصورة خطية inline

- **Ø** التجميعة nested table

تعتبر كجدول موجود في قاعة البيانات والاشارة الى اي عنصر في هذه التجميعة ايضا باستخدام الارقام السفلية ويتم تخزين البيانات في جدول تخزين منفصل.

- التعامل مع التجميعات :

قبل التعرف على كيفية التعامل مع التجميعات في لنموذج OO40 سوف نتعرف على كيفية التعامل معها من خلال SQL * PLUS والتعامل سوف يكون من جوانب الاضافة والحذف والتعديل وغيرها

--اولا :التعامل مع التجميعات في sql plus

أ -التجميعة varray من النوع البسيط

مثال/ نفرض انك تريد انشاء جدول الاقسام في مستشفى والذي سوف يحتوي على رقم القسم ، اسم القسم ، واسم القسم ، ومن ثم اسماء موظفين القسم .مع العلم ان اسماء الموظفين سوف تكون في تجميعةvarray .

نقوم اولا بانشاء التجميعة كمايلي:

```
Create type namev as varray(30) of varchar2(50);
/
```

حيث لو فرضنا ان اكبر عدد للموظفين هو 30 واكبر طول للاسم هو 50 .

ثم نقوم بأنشاء الجدول ونقوم بانشاءه كمايلي/

```
Create table deptv
(nodept number(5) primary key,
namedept varchar2(50),
emp namev);
```

ونقوم فيمايلي بتنفيذبعض اوامر sql على الجدول

١- الادراج insert :

```
Insert into deptv values(10,'medical',namev('ali','sami','fahad','fady'));
```

٢- التحديث update :

لتحديث العناصر في التجميعة يتطلب استخدام pl/sql ولايمكن تنفيذ ذلك من خلال sql القياسية

مثال:

```
Declare
  Editname namev;
  I   number:=1;
Begin
Select emp into editname
From deptv where nodept=10;
Loop
If (i=editname.count+1) then
Exit;
Elsif (editname(i)='sami') then
Editname(i):='mohammed';
End if;
i:=i+1;
end loop;
update deptv set emp=editname where nodept=10;
end;
```

شرح المثال السابق:
سوف يقوم بتغير اسم الموظف sami الذي قمنا بادخال وتبديلة الى mohammed وشرح الخطوات كمايلي
اولا قمنا بتعريف متغير editname من نفس نوع التجميعة namev وذلك لكي نقوم بتخزين المؤشر والذي يحتوي على اسماء الموظفين فيه ونعرف ايضا متغير I وهو من يستخدم كعداد.
ثم نقوم بعمل مؤشر لاستخراج اسماء الموظفين وهي كمايلي

```
Select emp into editname
From deptv where nodept=10;
```

يقوم هنا باستخراج اسماء الموظفين للقسم ١٠ وتخزين ناتج الاستعلام في المتغير editname والذي هو من نفس نوع التجميعة
ثم يبدأ حلقة ومن ثم يختبر هل I وصلت الى نهاية التجميعة اذا كان نعم قام بانهاء الاجراء واذا لم يصل الى نهاية التجميعة يختبر عنصر التجميعة الحالي هل هو يساوي sami ام لا اذا كان يساوي samiيقوم بتغير هذه القيمة الى mohammed ومن ثم يزيد العداد بواحد ومن ثم يعود من جديد الى ان يصل الى نهاية التجميعة وبعد الانتهاء من جميع العناصر يقوم بعمل التحديث للجدول

```
update deptv set emp=editname where nodept=10;
```

ومن ثم يقوم بانهاء الاجراء.

٢- الحذف delete(Trim) :

حذف عنصر من التجميعة يتطلب عمل اجراء pl/sql
والحذف في التجميعات varray يتم على اخر عنصر في التجميعة اي لو حذفنا عنصر واحد فانه يتم على اخر عنصر ولا يمكن تحديد العنصر
مثال:

```
Declare
Namedel namev;
Begin
Select emp into namedel
From deptv where nodept=10;
Namedel.trim(1);
update deptv set emp=namedel where nodept=10;
end;
/
```

٤- التحديث بالاضافة update(append)
هذا الامر يستخدم للاضافة
ألم يتبادر الى ذهنك كيف نضيف مزيدا من الموظفين الى القسم ١٠ يمكن ان تقول نستخدم الامر insert لكن هذا غير صحيح لاننا عندما نستخدم الامر insert وندخل رقم القسم ١٠ يظهر لنا خطاء لان حقل رقم القسم مفتاح رئسي
لذلك اذا اردنا اضافة المزيد من الموظفين اولا نقوم بعمل توسع extend للتجميعة لكي تسمح لنا باضافة عنصر جديد .ونستخدم الاجراء لذلك

ويكون الاجراء كمايلي:

```
Declare
Newname namev;
Begin
Select emp into newname
From deptv where nodept=10;
Newname.extend;
Newname(newname.last):='khaled';
update deptv set emp=newname where nodept=10;
end;
/
```

ب - التجميعة varray من النوع الشيء:
هذه التجميعة هي تجميعة معرفة بواسطة المستخدم

مثال ذلك :
لو اردنا انشاء جدول يحتوي على مسمى الوظيفة وفي حقل اخر نكّون تجميعة تحتوي على اسم الموظف وراتبة لجميع موظفين هذه الوظيفة

JOB_NAME	EMPLOYEE
manager	(ali,5000),(sami,6000),(fahad,4000)
Analysis	(laui,7500),(mohammed,7500)
Programming	(fady,8000),(saed,6000)

وبالتالي فإن الخطوة الاولى هي انشاء object كمايلي

```
Create type empobj
as object (nameemp varchar2(50),salary number(6));
/
```

ثم نقوم بانشاء التجميعة

```
Create type employeeobj as varray(20) of empobj;
/
```

لاحظ الفرق اننا استخدمنا object الذي انشئناه ولذلك سمي هذا النوع بهذا الاسم
وبعد انشاء التجميعة نقوم بإنشاء الجدول كمايلي:

```
Create table jobobj
(job_name varchar2(50),
employee employeeobj);
```

وبعد انشاء الجدول سوف نتعرف الان على كيفية التعامل مع هذا الجدول من خلال sql * plus

١ - الاضافة :
لاضافة صف إلى الجدول السابق نقوم بمايلي :

```
Insert into jobobj values
('manager',employeeobj (
                empobj('ali',5000),
                empobj('sami',6000),
                empobj('fahad',4000)));
```

وبهذا نكون قد اضفنا الصف الاول في الجدول السابق
ولو ذهبنا الى sql * plus وطلبنا منه مايلي

```
Select * from jobobj

JOB_NAME
-------------------------------------------
EMPLOYEEobj(NAMEEMP, SALARY)
-------------------------------------------------------
manager
EMPLOYEE1(EMPOBJ('ali', 5000), EMPOBJ('sami', 6000),
EMPOBJ('fahad', 4000))
```

ونلاحظ لقد تم اضافة الصف الجديد

شرح الامر

تجهيز الادخال

```
Insert into jobobj values
 ('manager',employeeobj (
        empobj('ali',5000),
        empobj('sami',6000),
        empobj('fahad',4000)));
```

اسم الوظيفة

اسم التجميعة

اسم object

٢- التحديث : يجب تطبيق قطعة pl/sql كمايلي:

```
Declare
Editname employeeobj;
Editobj empobj;
i number:=1;
Begin
Select employee into editname
From jobobj where job_name='manager';
Loop
Editobj:=editname(i);
If (i=editname.count) then
Exit ;
Elsif editobj.nameemp='sami' then
Editobj.salary:=10000;
Editname(i):=editobj;
End if;
i:=i+1;
End loop;
Update jobobj set employee=editname
Where job_name='manager';
End;
```

وبهذا يتم تعديل راتب الموظف الذي اسمه sami الموجود في قسم manager وجعل راتبه ١٠٠٠٠

٣- الحذف:

```
Declare
Editemp employeeobj;
begin
select Select employee into editemp
From jobobj where job_name='manager';
Editemp.trim(1);
Update jobobj set employee=editemp
Where job_name='manager';
```

وبهذا يتم حذف سجل واحد وهو اخر صف من التجميعة

ج- التجميعة NESTED TABLE من النوع البسيط:
تعتبر هذه التجميعة نفس التجميعة varray من النوع البسيط والتي نقوم بتعريفها بواسطة احد انواع البيانات الموجودة مثل number و varchar2 وسوف نستخدم هنا نفس المثال الذي استخدمناه في التجميعة varray من النوع الشيئ .
مثال/ نفرض انك تريد انشاء جدول الاقسام في مستشفى والذي سوف يحتوي على رقم القسم ، اسم القسم ، واسم القسم ، ومن ثم اسماء موظفين القسم .مع العلم ان اسماء الموظفين سوف تكون في تجميعةnested table
اولا نقوم بانشاء التجميعة وتكون كمايلي:

```
Create type namenested as table of varchar2(50);
/
```

ثم نقوم بأنشاء الجدول كمايلي:

```
Create table deptnested
(nodept number(5) primary key,
namedept varchar2(50),
emp namenested)
nested table emp store as nestedtablesimple;
```

لاحظ السطر الاخير الذي تم اضافنته
فيجب دائم وضعه اذا استخدمنا nested table مع تغير المكتوب باللون الاحمر والذي يمثل اسم الحقل الذي هو من النوع nested table.
والذي تحته خط يعتبر كأسم لهذا nested table

**التعامل مع الجدول السابق من خلال اوامر sql
١- الاضافة :

```
Insert into deptnested
values(1,'medical',namenested('ali','sami','fahad','fady'));
```

٢- التحديث update :

لتحديث العناصر في التجميعة يتطلب استخدام pl/sql ولايمكن تنفيذ ذلك من خلال sql القياسية
مثال:

```
Declare
  Editname namenested;
  I  number:=1;
Begin
Select emp into editname
From deptnested where nodept=1;
Loop
If (i=editname.count+1) then
Exit;
Elsif (editname(i)='sami') then
Editname(i):='mohammed';
End if;
i:=i+1;
end loop;
update deptnested set emp=editname where nodept=1;
end;
/
```

شرح المثال السابق:

سوف يقوم بتغير اسم الموظف samiالذي قمنا بادخال وتبديلة الى mohammed وشرح الخطوات كمايلي

اولا قمنا بتعريف متغير editname من نفس نوع التجميعة namenested وذلك لكي نقوم بتخزين المؤشر والذي يحتوي على اسماء الموظفين فيه ونعرف ايضا متغير I وهو يستخدم كعداد.

ثم نقوم بعمل مؤشر لاستخراج اسماء الموظفين وهي كمايلي

```
Select emp into editname
From deptnested where nodept=1;
```

يقوم هنا باستخراج اسماء الموظفين للقسم ١ وتخزين ناتج الاستعلام في المتغير editname والذي هو من نفس نوع التجميعة

ثم يبدأ حلقة ومن ثم يختبر هل I وصلت الى نهاية التجميعة اذا كان نعم قام بانهاء الاجراء واذا لم يصل الى نهاية التجميعة يختبر عنصر التجميعة الحالي هل هو يساوي sami ام لا اذا كان يساوي samiيقوم بتغير هذه القيمة الى mohammed ومن ثم يزيد العداد بواحد ومن ثم يعود

من جديد الى ان يصل الى نهاية التجميعة وبعد الانتهاء من جميع العناصر يقوم بعمل التحديث للجدول

```
update deptnested set emp=editname where nodept=1;
```

ومن ثم يقوم بانهاء الاجراء.

٣- التحديث بالاضافة update(append) :

اذا اردنا اضافة المزيد من الموظفين اولا نقوم بعمل توسع extend للتجميعة لكي تسمح لنا باضافة عنصر جديد .ونستخدم الاجراء لذلك

ويكون الاجراء كمايلي:

```
Declare
Newname namenested;
Begin
Select emp into newname
From deptnested where nodept=1;
Newname.extend;
Newname(newname.last):='khaled';
update deptnested set emp=newname where nodept=1;
end;
/
```

٤-الحذف :

هناك طريقتان الاولى باستخدام trim وهذه الطريقة تحذف اخر صف في التجميعة ولايمكن اختيار اي عنصر في التجميعة:

```
Declare
Namedel namenested;
Begin
Select emp into namedel
From deptnested where nodept=1;
Namedel.trim(1);
update deptnested set emp=namedel where nodept=1;
end;
/
```

اما الطريقة الثانية فهي باستخدام الامر delete(m) وهذه الطريقة لا تحذف من الاخير بل يتم اختيار اي العناصر الذي تريد حذف فلو وضعنا m=2 وكانت المدخلات الجدول كما فعلا سابقا سيتم حذف العنصر الثاني والذي هو الموظف sami

```
Declare
Namedel namenested;
Begin
Select emp into namedel
From deptnested where nodept=1;
Namedel.delete(2);
update deptnested set emp=namedel where nodept=1;
end;
/
```

وهذه الطريقة(طريقة delete) هي من ابرز الفوارق بين varray و nested table حيث varray لا يمكنها حذف اي عنصر ولكن تحذف فقط العنصر الاخير بعكس nested table

د- التجميعة NESTED TABLE من النوع الشيئ:
كما لاحظنا ان التجميعة varray يتم انشاءها من النوع الشيء فإن NESTED TABLE يتم إنشاءه ايضا من النوع الشيء المعرف بواسطة المستخدم.
سوف نستخدم لشرح هذا المثال نفس المثال الذي استخدمناه في varray من النوع object

مثال ذلك :
لو اردنا انشاء جدول يحتوي على مسمى الوظيفة وفي حقل اخر نكّون تجميعة تحتوي على اسماء الموظف وراتبة لجميع موظفين هذه الوظيفة

اول خطوة هي انشاء الشيء empobj والذي تم انشاءه سابق عندما قمنا بشرح vaaray من النوع الشيء واذا كنت لم تقم بانشاءه فهذا الكود :

```
Create type empobj
as object (nameemp varchar2(50),salary number(6));
  /
```

ثاني خطوة هي إنشاء التجميعة nested table باستخدام الشيء empobj كمايلي

```
Create type empnestedobj as table of empobj;
  /
```

ثالث خطوة نقوم بإنشاء الجدول كمايلي:

```
Create table jobnested
(job_name varchar2(50),
employee empnestedobj)
nested table employee store as nestedtablesimple;
```

* التعامل مع الجدول السابق من خلال اوامر sql
١- الاضافة :
لكي تقوم بادراج بيانات في الجدول السابق نقوم بمايلي:

```
Insert into jobnested
Values('manager',
       empnestedobj(empobj('ali',6000),
                    empobj('sami',7000),
                         empobj('fahad',6500)));
```

٢- التحديث : يتم التحديث باستخدام pl/sql كمايلي:

مثلا لتغير راتب الموظف ali الموظف الاداري اي في مسمى وظيفتها manager الى ٩٠٠٠

```
Declare
Editsal empnestedobj;
Editempobj empobj;
i number:=1;
Begin
Select employee into editsal
From jobnested where job_name='manager';
Loop
Editempobj:=editsal(i);
If (i=editsal.count) then
Exit;
Elsif (editempobj.nameemp='fahad') then
Editempobj.salary:=9900;
Editsal(i):=editempobj;
End if;
i:=i+1;
end loop;
update jobnested set employee=editsal where job_name='manager';
end;
/
```

٣- التحديث بالاضافة update(append) :-

```
Declare
Editemp empnestedobj;
Begin
Select employee into editemp
From jobnested where job_name='manager';
Editemp.extend;
Editemp(editemp.last):=empobj('mohammed',7000);
update jobnested set employee=editemp where job_name='manager';
end;
/
```

٤- الحذف delete(trim) :

هناك طريقتان الاولى باستخدام trim وهذه الطريقة تحذف اخر صف في التجميعة ولايمكن اختيار اي عنصر في التجميعة:

```
declare
empdel empnestedobj;
begin
Select employee into empdel
From jobnested where job_name='manager';
Empdel.trim(1);
update jobnested set employee=empdel where job_name='manager';
end;
/
```

اما الطريقة الثانية فهي باستخدام الامر delete(m) وهذه الطريقة لا تحذف من الاخير بل يتم اختيار اي العناصر الذي تريد حذف فلو وضعنا m=2 وكانت المدخلات الجدول كما فعلا سابقا سيتم حذف العنصر الثاني وهو الموظف sami وفيمايلي الكود

الطريقة الثانية

```
declare
empdel empnestedobj;
begin
Select employee into empdel
From jobnested where job_name='manager';
Empdel.delete(2);
update jobnested set employee=empdel where job_name='manager';
end;
/
```

--ثانياً :التعامل مع التجمعات من خلال النموذج OO4O:

أن من أحد الانواع الجديدة التي أدخلها النموذج OO4O هو النوع oraobject و النوع oracollection

ومن أهم مواصفات النوع oracollection مايلي:

١- المواصفة maxsize حيث تكون كمايلي:
بالنسبة للتجميعة varray تحتوي هذه المواصفة على اقصى حجم للتجميعة والحجم المستخدم عند تعريف التجميعة
بالنسبة للتجميعة nested table تحتوي على الحجم الحالي للتجميعة والسبب في ان maxsize تمثل الحجم الحالي هو ان التجميعة nested table ليس لها حد أقصى.

٢- المواصفة size :
وهذه المواصفة تحتوي على الحجم الحالي للتجميعة
ولكن يجب ملاحظة ان هذه المواصفة size تحسب العناصر المحذوفة من التجميعةايضا ضمن الحجم الحالي.

٣- المواصفة tablesize :
وهذه المواصفة تحتوي على الحجم الحالي للتجميعة nested table وتستثني السجلات التي تم حذفها delete

٤- الموصفة APPEND:
تؤدي هذه الوسيلة إلى توسيع التجميعة بعنصر واحد وتضيف القيمة الجديدة الى نهاية التجميعة.
ومن الوسائل الهامة للنوع oracollection مايلي:

١- الوسيلة delete :
هذه الوسيلة تطبق فقط على التجميعات nested table وتسمح لك بحذف عناصر في التجميعة nested table بتمرير الرقم السفلي الذي تريد حذفه

٢- الوسيلة Trim(m) :
تحذف العدد m من نهاية التجميعة.

٣- الوسيلة createoraobject :
وهي وسيلة للـ oradatabase
تستخدم لانشاء شيء (object) جديد والهيئة العامة:

```
Set obj=database.createoraobject("OBJECT")
Set coll=database.createoraobject("COLLECTION")
```

التعامل مع التجميعات:

١- التعامل مع التجميعة varray من النوع البسيط:

ولكي يتم الاستيعاب اكثر سوف نستخدم نفس الجدول الذي قمنا بانشاءه سابقا وفي التعامل التجميعة varray من النوع البسيط في sql * plus واسم الجدول هو deptv

وايضا التجميعة التي قمنا بإنشاءها سابقا واسنها واذا كنت لم تقم بانشاءه namev واذا لم تكن انشئتهن تقوم بمايلي :

اولا انشاء التجميعة كمايلي:

```
Create type namev as varray(30) of varchar2(50);
/
```

ثانياً انشاء الجدول كمايلي:

```
Create table deptv
(nodept number(5) primary key,
namedept varchar2(50),
emp namev);
```

ثم نبدأ بالتعامل مع الجدول كمايلي

أ- الاضافة insert :

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd","scott/tiger", 0&)
SQL = "select * from deptv"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = database.CreateOraObject("NAMEV")
dept(1) = "هادي محمد"
dept(2) = "فهد احمد"
dept(3) = "نايف محمد"
dept(4) = "محمد ابراهيم"
dyn.AddNew
dyn.Fields("nodept").Value = "30"
dyn.Fields("namedept").Value = "الجراحة"
dyn.Fields("emp").Value = dept
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

وبهذا تمت الاضافة الجديدة لاحظ انه يمكنك ان تزيد عناصر التجميعة dept(1),dept(2),dept(3)……dept(n)

ب – التحديث update:

مثلا بناء على المدخلات السابقة لو اردنا تبديل اسم "فهد احمد" الموظف في قسم الجراحة إلى "فهد طارق"

فيتم ذلك كمايلي:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("mohammed", "scott/tiger", 0&)
SQL = "select * from deptv where nodept=30"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
For i = 1 To dept.Size
If dept(i) = "فهد احمد" Then
dyn.Edit
dept(i) = "فهد طارق"
dyn.Update
MsgBox "تم التعديل بنجاح"
Exit Sub
End If
Next i
MsgBox " لم يتم التعديل بنجاح"
```

لاحظ هنا شيء واحد هو الذي يمكن ان يخيفك وهو

Set dept = dyn.Fields("emp").Value

لاحظ انه لم يتم اسناد المتغير dept وهو من نوع التجميعة الى مسمى التجميعة مثل عملية الادخال والسبب هنا نود معرفة الحقل الذي هو من نوع التجميعة وهذا ماحدث حيث اخبرناه باسم الحقل.

ج – التحديث بالاضافة update(append) :
مثال ذلك لو اردنا اضافة موظف جديد الى قسم الجراحة فيتم ذلك كمايلي :
الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("mohammed", "scott/tiger", 0&)
SQL = "select * from deptv where nodept=30"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
dyn.Edit
dept.Append "علي عبدالله"
dyn.Update
MsgBox "تم الاضافة بنجاح"
```

ولاحظ هنا اننا استخدمنا الامر append لكي يقوم بعمل توسيع للتجميعة ويضيف الى اخرها الاسم الجديد
ويمكنك ان تضع هذا الكود داخل زر وتضع مربع نص ومن مربع النص هذا تدخل اسماء الموظفين ويكون الكود نفس السابق مع تغير سطر الاضافة كمايلي:

```
dept.Append  Text1.text
```

وبهذا يتم اضافة الاسم الموجود في مربع النص كلما ضغطت على زر الاضافة.

د – الحذف delete(trim)
كما تعلمنا سابقا ان التجميعة varray يتم الحذف بها من اخر عنصر ولا يمكن اي العناصر التي تريدحذفها كما في nested table مثلا لحذف اخر اسم من التجميعة نقوم بمايلي:
الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("mohammed", "scott/tiger", 0&)
SQL = "select * from deptv where nodept=30"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
dyn.Edit
dept.Trim 2
dyn.Update
MsgBox "تم الحذف بنجاح"
```

لاحظ في السطر **dept.Trim 2**
اننا وضعنا 2 بعد trim وهذا لايعني اننا نريد حذف اخر عنصرين ولكن حذف العنصر الاخير فقط والسبب في ذلك ان الوسيلة trim تطرح واحد من العدد الممرر.

٢- التعامل مع التجميعة varray من النوع الشيء:
سوف نستخدم نفس الجدول الذي استخدمناه في الامثلة التي تتعامل مع التجميعة varray من النوع الشيء في sql*plus وللتذكير به.
الجدول اسمه jobobj وهو جدول يحتوي على مسمى الوظيفة في حقل وفي حقل اخر نكّون تجميعة تحتوي على اسم الموظف وراتبة لجميع موظفين هذه الوظيفة
والتجميعة varrayهي employeeobj

واذا كنت لم تقم بانشاء هذا الجدول والتجميعة سابقا عند التعامل مع التجميعة varray من النوع الشيء فقم بإنشاءها كمايلي
الخطوة الاولى هي انشاء object كمايلي

```
Create type empobj
as object (nameemp varchar2(50),salary number(6));
/
```

ثم نقوم بانشاء التجميعة

```
Create type employeeobj as varray(20) of empobj;
/
```

لاحظ الفرق اننا استخدمنا object الذي انشئناه ولذلك سمي هذا النوع بهذا الاسم
وبعد انشاء التجميعة نقوم بإنشاء الجدول كمايلي:

```
Create table jobobj
(job_name varchar2(50),
employee employeeobj);
```

ثم نبدأ بالتعامل مع الجدول كمايلي :

أ-الاضافة insert :

الكود :

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobobj"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = database.CreateOraObject("EMPLOYEEOBJ")
Set obj = database.CreateOraObject("EMPOBJ")
obj.nameemp = "محمد ابراهيم"
obj.salary = 7000
dept(1) = obj
obj.nameemp = "حمد ناصر"
obj.salary = 8500
dept(2) = obj
dyn.AddNew
dyn.Fields("job_name").Value = "المبرمجون"
dyn.Fields("employee").Value = dept
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

وبهذا يتم وظيفة المبرمجون ومعها الموظفين بهذه الوظيفة.

ب- التحديث update :

مثلا لو اردنا تعديل راتب الموظف "محمد ابراهيم" والذي يعمل في وظيفة "مبرمج" من ٧٠٠٠ إلى ٧٢٠٠ فيصبح الكود كمايلي:

الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobobj where JOB_NAME='المبرمجون'"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("employee").Value
For i = 1 To dept.Size
Set obj = dept(i)
If obj.nameemp = "محمد ابراهيم" Then
dyn.Edit
obj.salary = 7200
dyn.Update
MsgBox "تم التعديل بنجاح"
Exit Sub
End If
Next i
MsgBox "لم يتم تعديل البيانات"
```

وبهذا يتم تعديل راتب الموظف محمد ابراهيم

ج- التحديث بالاضافة update(append) :

مثال لو اردنا اضافة موظف جديد إلى قسم المبرمجون مثلا الموظف "فهد محمد" وراتبة "٦٥٠٠"
فيصبح الكود كمايلي:

الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobobj where JOB_NAME='المبرمجون'"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("employee").Value
Set obj = database.CreateOraObject("EMPOBJ")
obj.nameemp = "فهد محمد"
obj.salary = 6500
dyn.Edit
dept.Append obj
dyn.Update
MsgBox "تم اضافة الموظف الجديد بنجاح"
```

ويهذا يتم اضافة الموظف الجديد مع راتبة.

د- الحذف delete(trim) :

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobobj where JOB_NAME='المبرمجون'"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("employee").Value
dyn.Edit
dept.Trim 2
dyn.Update
MsgBox "تم حذف الموظف بنجاح"
```

لاحظ في السطر **dept.Trim 2**
اننا وضعنا 2 بعد trim وهذا لايعني اننا نريد حذف اخر عنصرين ولكن حذف العنصر الاخير فقط والسبب في ذلك ان الوسيلة trim تطرح واحد من العدد الممرر.

٣- التعامل مع التجميعة nested table من النوع البسيط:
سوف نستخدم نفس الجدول الذي استخدمناه في nested table من النوع البسيط عندما تعاملنا معها بـ sql*plus وللتذكير به:
هو جدول الاقسام في مستشفى والذي سوف يحتوي على رقم القسم ، اسم القسم ، واسم القسم ، ومن ثم اسماء موظفين القسم .مع العلم ان اسماء الموظفين سوف تكون في تجميعةnested table

اذا لم تقوم بانشاء التجميعة قم بإنشاءها كمايلي:

```
Create type namenested as table of varchar2(50);
/
```

واذا لم تقوم بأنشاء الجدول فهو كمايلي:

```
Create table deptnested
(nodept number(5) primary key,
namedept varchar2(50),
emp namenested)
nested table emp store as nestedtablesimple;
/
```

وللتعامل مع الجدول السابق من خلال OO40 كمايلي:
أ – الاضافة insert:
الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from deptnested"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = database.CreateOraObject("NAMENESTED")
dept(1) = "محمد ابراهيم"
dept(2) = "حمد ناصر"
dept(3) = "خالد صالح"
dyn.AddNew
dyn.Fields("nodept").Value = 15
dyn.Fields("namedept").Value = "الجراحة"
dyn.Fields("emp").Value = dept
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

ب- التحديث UPDATE :

مثلا لو اردنا تعديل اسم الموظف "حمد ناصر" الموجود في قسم الجراحة الى "حمد سليمان" فيتم ذلك كمايلي:
الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from deptnested where NODEPT=15"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
For i = 1 To dept.Size
If dept(i) = "حمد ناصر" Then
dyn.Edit
dept(i) = "حمد سليمان"
dyn.Update
MsgBox "تم التعديل بنجاح"
Exit Sub
End If
Next i
MsgBox "البيانات لم يتم تعديلها"
```

وبعد ذلك يتم تعديل اسم الشخص.

ج- التحديث بالاضافة update(Append):

مثال: نريد اضافة اسم موظف جديد الى المستشفى وهو "فهد محمد" وتم تعيينه في قسم الجراحة فيتم ذلك كمايلي:

الكود

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from deptnested where NODEPT=15"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
dyn.Edit
dept.Append "فهد محمد"
dyn.Update
MsgBox "تم الاضافة بنجاح"
```

وبهذا الكود يتم اضافة الموظف "فهد محمد" إلى قسم الجراحة في المستشفى.

د- الحذف delete :

هنا يتم حذف العنصر الذي تريد من التجميعة مثل العنصر الثالث او الرابع او اي عنصر بعكس trim التي تحذف العنصر الاخير.

فمثلا:اذا اردنا حذف العنصر الثالث من الموظفين الذين يعملون في قسم الجراحة من التجميعة نقوم بمايلي:

الكود

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from deptnested where NODEPT=15"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
dyn.Edit
dept.Delete 3
dyn.Update
MsgBox "تم الحذف بنجاح"
```

وبعد ذلك يتم حذف العنصر الثالث من التجميعة ولاحظ الرقم ٣ انه هو رقم العنصر المحذوف.

د – الحذف trim :

هذا الطريقة تقوم بحذف اخر عنصر من التجميعة ولا يتم اختيار العنصر الذي تريد حذفه ويتم ذلك كمايلي:

الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from deptnested where NODEPT=15"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
dyn.Edit
dept.Trim 2
dyn.Update
MsgBox "تم الحذف بنجاح"
```

وكما قلنا سابقا ان

لاحظ في السطر **dept.Trim 2**

اننا وضعنا 2 بعد trim وهذا لايعني اننا نريد حذف اخر عنصرين ولكن حذف العنصر الاخير فقط والسبب في ذلك ان الوسيلة trim تطرح واحد من العدد الممرر.

مـــــثـــال:
هذا المثال شامل سوف نستخدم فيه ٣ عمليات هي التعديل والتعديل بالاضافة والحذف في نفس الوقت والمثال هو
قم بتعديل اسم الموظف الموجود في قسم العيون وهو "ناصر علي" الى "ناصر محمد" ثم
قم بنقله من قسم "العيون" الى قسم "الجراحة".
لو فرضنا ان هناك قسم "العيون" ويوجد به عدد من الموظفين مثلا نقوم بإنشاء اولا قسم المهندسون

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobobj"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = database.CreateOraObject("EMPLOYEEOBJ")
Set obj = database.CreateOraObject("EMPOBJ")
obj.nameemp = "سامي احمد"
obj.salary = 6500
dept(1) = obj
obj.nameemp = "خالد صالح"
obj.salary = 8500
dept(2) = obj
dyn.AddNew
dyn.Fields("job_name").Value = "المهندسون"
dyn.Fields("employee").Value = dept
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

الان نعود الى الى مثال وهو تعديل اسم الموظف "ناصر علي" الموجود في قسم "العيون" الى "ناصر محمد"
ثم نقوم بنقله من قسم"العيون" الى قسم "الجراحة".
فيتم ذلك كمايلي

كود التعديل:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from deptnested  where NODEPT=20"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
For i = 1 To dept.Size
If dept(i) = "ناصر علي" Then
dyn.Edit
dept.Delete i
dyn.Update
End If
Next i
dyn.Close
SQL = "select * from deptnested where NODEPT=15"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("emp").Value
dyn.Edit
dept.Append "ناصر محمد"
dyn.Update
MsgBox "تم النقل والتعديل بنجاح"
```

٣- التعامل مع التجميعة nested table من النوع الشيء:
سوف نستخدم نفس المثال الذي استخدمناه مع nested table عندما تعاملنا معها من خلال sql * plus وهي كمايلي:
لو اردنا انشاء جدول يحتوي على مسمى الوظيفة وفي حقل اخر نكّون تجميعة تحتوي على اسماء الموظف وراتبة لجميع موظفين هذه الوظيفة
اذا كنت لم تقم بانشاء object او التجميعة او الجدول فيتم انشاءها كمايلي اما اذا كنت انشتها سابق عندما تعاملنا مع هذه الطريقة ب sql * plus فلا يلزم انشاءها.
اول خطوة هي انشاء الشيء empobj والذي تم انشاءه سابق عندما واذا كنت لم تقم بانشاءه فهذا الكود :

```
Create type empobj
as object (nameemp varchar2(50),salary number(6));
  /
```

ثاني خطوة هي إنشاء التجميعة nested table باستخدام الشيء empobj كمايلي

```
Create type empnestedobj as table of empobj;
  /
```

ثالث خطوة نقوم بإنشاء الجدول كمايلي:

```
Create table jobnested
(job_name varchar2(50),
employee empnestedobj)
nested table employee store as nestedtableobj;
```

** نبدأ التعامل مع الجدول السابق من خلال OO40 :
١- الاضافة insert :
مثلا لو اردنا اضافة قسم "المبرمجون"
والموظفون هم :
"محمد ابراهيم" وراتبة ٦٠٠٠
"حمد صالح" وراتبة ٨٥٠٠
"ناصر فهد" وراتبه ٧٥٠٠
فيصبح الكود كمايلي:

الكود :

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobnested"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = database.CreateOraObject("EMPNESTEDOBJ")
Set obj = database.CreateOraObject("EMPOBJ")
obj.nameemp = "محمد ابراهيم"
obj.salary = 6000
dept(1) = obj
obj.nameemp = "حمد صالح"
obj.salary = 8500
dept(2) = obj
obj.nameemp = "ناصر فهد"
obj.salary = 7500
dept(3) = obj
dyn.AddNew
dyn.Fields("job_name").Value = "المبرمجون"
dyn.Fields("employee").Value = dept
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

وبهذا تتم الاضافة.

٢- التعديل:

مثال/ ثم بتعديل راتب الموظف "محمد ابراهيم" من ٦٠٠٠ الى ٦٥٠٠

الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobnested"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("EMPLOYEE").Value
For i = 1 To dept.Size
Set obj = dept(i)
If obj.nameemp = "محمد ابراهيم " Then
dyn.Edit
obj.salary = 6500
dyn.Update
MsgBox "تم التعديل بنجاح"
Exit Sub
End If
Next i
MsgBox "لم يتم التعديل"
```

وبهذا يتم التعديل على راتب الموظف محمد ابراهيم.

٣- التعديل بالاضافة update(Append) :

مثلا لكي تضيف الموظف تركي بدر والذي راتبه ٧٥٠٠ الى قسم المبرمجون يكون الكود كمايلي:

الكود:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobnested where job_name='المبرمجون'"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("EMPLOYEE").Value
Set obj = database.CreateOraObject("EMPOBJ")
obj.nameemp = "تركي بدر"
obj.salary = 7500
dyn.Edit
dept.Append obj
dyn.Update
MsgBox "تمت الاضافة بنجاح"
```

٤- الحذف

**الطريقة الاولى delete:

مثلا لحذف العنصر الثالث في تجميعة المبرمجون نقوم بمايلي:

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobnested where job_name='المبرمجون'"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("EMPLOYEE").Value
dyn.Edit
dept.Delete 3
dyn.Update
MsgBox "تم الحذف بنجاح"
```

**الطريقة الثانية trim:
لحذف اخر عنصر في تجميعة المبرمجون

```
Dim ses As OraSession
Dim database As OraDatabase
Dim dyn As OraDynaset
Dim dept As OraCollection
Dim obj As OraObject
Set ses = CreateObject("oracleinprocserver.xorasession")
Set database = ses.OpenDatabase("ddd", "scott/tiger", 0&)
SQL = "select * from jobnested where job_name='المبرمجون'"
Set dyn = database.CreateDynaset(SQL, 0&)
Set dept = dyn.Fields("EMPLOYEE").Value
dyn.Edit
dept.Delete 3
dyn.Update
MsgBox "تم الحذف بنجاح"
```

وتم شرح الفرق بين هذين النوعين سابقا

التعامل مع الاجراءات المخزنة من خلال البارامترات:

قبل البدء بشرح طرق الاتصال بالاجراءت سوف ناخذ شرح عن pl/sql :

المعاملات الحسابية المستخدمة

فيمايلي اغلب المعاملات الحسابية المستخدمة بكثرة في لغة pl/sql واذا كنت معتاد على لغة برمجة فلن تكون هذا الشي جديد عليك

+	الجمع
*	الضرب
**	الاس
-	الطرح
/	القسمة

المعاملات العلاقية :

< >	لايساوي
= ^	لايساوي
>	اكبر من
!=	لايساوي
<	اقل من

انواع البيانات شائعة الاستخدام :

١- varchar2

هذا النوع متغير الطول ويشتمل على الاحرف الابجدية والارقام

```
X    varchar2(20)
```

حيث الموجود داخل القوسين هو length الطول
ومن الممكن اعطاءه قيمة ابتدائية كمايلي

```
X   varchar2(20)='hamad'
```

٢- number

يستخدم لتمثيل البيانات الرقمية وتكون صيغة الاعلان كمايلي:

```
Num  number(s)
```

S هي عدد الارقام(الخانات) وتاخذ قيمة بين 1 إلى 38

ويمكن ايضا تعريف اي متغير رقمي من النوع العشري كمايلي:

```
Num  number(s,p)
```

حيث s عدد خانات الرقم الصحيح وايضا العشري
اما p فهي عدد المنازل(الخانات) بعد الفاصلة مثال

```
Num number(12,2)
```

معنى هذا ان الرقم num مكون من ١٠ ارقام صحيحة وؤقمين بعد الفاصلة وبذلك بكون المجموع 12

٣- date :
يستخدم هذا المتغير لتخزين قيم التواريخ مثل

```
Date_brith  date;
```

في الوضع الافتراضي يعرض اوراكل قيمة التاريخ بالشكل
DD-MON-YY

٤- Boolean :
منطقي true او false

نبدأ الان بمكونات pl/sql :
برامج pl/sql تتم كتابتها في كتل من اوامر البرمجة تحتوي على مقاطع منفصلة للاعلان عن المتغيرات واوامر البرامج ومعالجة الاستثناءت(الاخطاء) .
ومن الممكن تخزبن الاجراء في قاعدة البيانات كبرنامج فرعي له اسم محدد او كتابتها مباشرة في sql * plus ككتله مجهولة.

وكتلة البرنامج كمايلي:
سنبدا اولا كتابة الاجراء مباشرة في sql * plus وهي كمايلي:

```
DECLARE
        هنا توجد تعريفات المتغيرات والموشرات
BEGIN
  جسم البرنامج

EXCEPTION
رموز معالجة الاخطاء
END:
```

مع ملاحظة مايلي:
ان قسم declare وقسم exception هما اختيارين اي لايشترط وجودهما
اي اذا كان لايوجد لديك تعريف متغيرات لاتستخدم declare واذا كنت لان تتعامل مع الاخطاء لاتستخدم exception

---

طرق الاسناد
مثل اذا اردت ان تقول ان قيمة i=5 فيتم ذلك كمايلي:

```
i:=5;
```

يجب وضع النقطتين قبل =

سوف ناخذ مثال على ذلك
يوجد ضمن اوامر sql الامر DBMS_OUTPUT.PUT_LINE
يستخدم لكي تعرض النتيجة في sql * plus والصيغة العامة له :

```
DBMS_OUTPUT.PUT_LINE(massege)
```

حيث massage هي النص او الشي الذي تريد عرضة
تم شرح هذا الامر لكي نبدأ به ونستخدمه لفهم اوامر ال pl/sql لكن في المستقبل سوف تعرف ان هذا الامر لايهمك كثيرا.مثال :
نريد طباعة "ARABTEAM2000" على الشاشة sql * plus يتم ذلك كمايلي:

```
SET SERVEROUTPUT ON;
BEGIN
DBMS_OUTPUT.PUT_LINE('ARABTEAM2000');
End;
```

جرب هذا ولاحظ النتائج ولعلك تتسال عن سبب وجود السطر الاول
السطر الاول يخبر sql * plus بأن يكتب كل مايعود به المخدم.
ويكفي كتابته مرة واحدة عندما تدخل sql * plus
مثال اخر باستخدام المتغيرات

```
Declare
i   number(5);
BEGIN
i:=5;
DBMS_OUTPUT.PUT_LINE('i = ' || i);
END;
```

جرب هذا الكود ولاحظ النتائج
فائدة || الموجدة ضمن عملة الطباعة هي للوصل بين التعبيرين

اوامر اللغة :

أ- الشرط عبارة if then

تستخدم هذه العبارة مثل اي العبارت الشرطية في لغة سي او سي++ او فيجوال بيسك وغيرها ، ولها استخدمات عديدة وسوف نعرف كيف نستخدمها مقدما مع حقول قواعد البيانات وذلك بعد اخذ المؤشرات

الصيغة العامة لها كمايلي:

```
IF  conditonal    THEN
جواب الشرط
ELSE
جواب الشرط اذا كان خطأ
END IF
```

حيث conditional هي الشرط مثال على ذلك :

```
Declare
i  number(5);
BEGIN
i:=5;
IF i=5 then
DBMS_OUTPUT.PUT_LINE('i = ' || i);
ELSE
DBMS_OUTPUT.PUT_LINE('i not eqal 5 ');
END IF;
END;
```

الشرط باستخدام اكثر من شرط

```
Declare
i  number(5);
BEGIN
i:=5;
IF i>1 then
DBMS_OUTPUT.PUT_LINE(i  || ' > 1');
ELSIF i<1 then
DBMS_OUTPUT.PUT_LINE(i  || ' < 1');
ELSIF i=1 then
DBMS_OUTPUT.PUT_LINE(i  || ' = 1');
END IF;
END;
```

لكن لاحظ هنا فقط كتابة elsif ولاتخطي فيها فهي ليست elseif

ب- الــتكرار :
ويوجد عدة اوامر للتكرار وهي:
١- loop-exit-end
وهنا لابد من وضع شرط لأنهاء الحلقة
نأخذ مثال بسيط على ذلك

```
Declare
i   number(5);
BEGIN
i:=1;
LOOP
IF i>10 then
EXIT;
END IF;
DBMS_OUTPUT.PUT_LINE('i =' || i);
i:=i+1;
End loop;
END;
/
```

شرح الكود السابق
السطر الثاني : تعريف متغير من نوع رقم
السطر الثالث :البداية
السطر الرابع : اعطاء المتغير قيمة ابتدائية وهي i=1
السطر الخامس : شرط الانهاء
السطر السادس : الانهاء اذا كان i>10 ولا يكمل
السطر السابع :انها if
السطر الثامن : طباعة i
السطر التاسع :زيادة قيمة i بواحد
السطر العاشر : نهاية الحلقة
وبعد كتابة هذا الكود يكون الناتج كمايلي:

i =1
i =2
i =3
i =4
i =5
i =6
i =7
i =8
i =9
i =10

٢- EXIT - END LOOP - WHEN

```
Declare
i   number(5);
BEGIN
i:=1;
LOOP
EXIT WHEN i>10;
DBMS_OUTPUT.PUT_LINE('i =' || i);
i:=i+1;
End loop;
END;
/
```

ويكون الناتج نفس السابق لكن لاحظ استخدم شرط الانهاء

```
EXIT WHEN i>10;
```

٣- WHILE - LOOP - END

```
Declare
i   number(5);
BEGIN
i:=1;
WHILE i <= 10 LOOP
DBMS_OUTPUT.PUT_LINE('i =' || i);
i:=i+1;
End loop;
END;
/
```

ويكون الناتج نفس المثال السابق

٤- FOR - IN - LOOP - END :

وهذه ايضا طريقة اخرى لاستخدام حلقات التكرار وهي نفس عمل اسلوب حلقات for في اي لغة برمجة

والصيغة العامه لها هي على الاتي

FOR i IN البداية.. النهاية LOOP

الجمل المراد تكرارها

LOOP END

مثـــــــــــــال:

```
Declare
i   number(5);
BEGIN
FOR  i   IN  1..10 LOOP
DBMS_OUTPUT.PUT_LINE('i =' || i);
End loop;
END;
/
```

وسوف يكون الناتج كمايلي(نفس السابق):

i١=
i =2
i =3
i =4
i =5
i =6
i =7
i =8
i =9
i= 10

ج/ المؤشرات CURSORS

هذا الدرس من اهم المواضيع في pl/sql
تستخدم pl/sql المؤشرات cursors لأدارة عبارات التحديد select في لغة sql وكما لاحظنا الاوامر السابقة مثل if والتكرار لم نستخدمها مع بيانات الجداول المخزنه ولعمل ذلك لابد من استخدام هذه المؤشرات.
وهناك نوعين من المؤشرات هي الضمنية والصريحة وسوف نتطرق لك واحد بالتفصيل والامثلة اللازمة.
أ- المؤشرات الصريحه :
يتم تعريف هذا النوع من المؤشرات كجزء من الاعلان declare ويجب ان تشتمل عبارة sql المعرفه على عبارة التحديد select فقط حيث لايمكن استخدام الكلمات الاساسية insert,update,delete
وعند استخدام المؤشرات الصريحه دائما ماستكتب اربعة مكونات كمايلي:
١- يتم تعريف المؤشر في الجزء declare
٢- يتم فتح المؤشر بعد عبارة begin

الصيغة العامة لتعريف المؤشر الصريح كمايلي:

DECLARE
CURSOR اسم المؤشر IS
الاستعلام

تقوم باستبدال اسم المؤشر باسم مؤشر حقيقي
وتقوم بوضع جملة الاستعلام select في مكان الاستعلام

ولكي تقوم بفتح هذا المؤشر وتستخدمه نقوم بفتحه باستخدام الامر open كمايلي:
OPEN اسم المؤشر

وبعد فتح المؤشر تقوم باسترجاع او تحميل البيانات سطر(سجل) واحد من المؤشر الذي تم تعريفه باستخدام الامر FETCH كمايلي:
FETCH اسم المؤشر INTO متغير١،متغير٢،.....

ومعنى هذا اي قم باسترجاع البيانات من المؤشر المعطى اسمه وحملها into الى المتغيرات كع ملاحظة ان عدد المتغيرات يساوي عدد الحقول الموجودة في استعلام المؤشر.

وبعد الانتهاء من اجراء العمليات على المؤشر يجب عليك اغلاقه ويتم اغلاقه كمايلي:
close cursor_name

مثال على طريقة تعريف مؤشر :
افرض انه لدينا هذا الجدول

age	name	no
23	mohammed	111
22	talal	222
24	majed	333

اولا قم بانشاء هذا الجدول كمايلي:

```
create table stud(
no number(4),
name varchar2(40),
age number(2));
```

ثانيا قم بادخال البيانات السابقة في هذا الجدول كمايلي:

```
insert into stud values(111,'mohammed',23);
insert into stud values(222,'talal',22);
insert into stud values(333,'majed',24);
```

ثم قم بتنفيذ مايلي

```
set serveroutput on;
DECLARE
name_stu varchar2(40);
CURSOR  name_student IS
select name from stud
where no=111;
BEGIN
OPEN name_student;
FETCH name_student INTO name_stu;
DBMS_OUTPUT.PUT_LINE(name_stu);
CLOSE name_student;
END;
/
```

بعد التنفيذ سوف يظهر لك الناتج كمايلي:

mohammed

وهذا الشي صحيح
لاحظ اننا اتبعنا نفس الخطوات التي ذكرناه لكي نتعامل مع مؤشر صريح

لاحظنا في المثال السابق ان الاستعلام في cursor سوف يعود بسجل واحد لكن ماذا يحدث لو اعاد المؤشر اكثر من سجل واردنا المرور على كافة السجلات ؟

لحل السؤال السابق لابد من استخدام حلقة بها شرط وهذا هو هل سجلات المؤشر انتهت ام لا ونعرف ذلك من خلال خاصية found للمؤشر كمايلي:

found%mycur

حيث :

mycur : هي اسم المؤشر.

% : توضح انا مايلي اسم المؤشر هي احد خصائصه.

found : خاصية التي من خلالها نعرف هل تم الانتهاء من جميع السجلات ام لا

مثال :

اسم الطالب	كود المقرر	الدرجة	النتيجة
NO_STU	SUBJECT	MARK	RESULT
111	216CS	88	
222	225CS	75	
333	225CS	40	

نريد انشاء اجراء يقوم بالمرور على الجدول وينظر الى درجة الطالب اذا كان ناجح في المقرر ام لا فاذا كان mark اكبر او يساوي ٥٠ ضع قيمة true في حقل result والا ضع قيمة false في حقل result

نقوم اولا بانشاء هذا الجدول :

```
create table stu_study(
NO_STU number(4),
SUBJECT varchar2(8),
MARK number(3),
RESULT varchar2(20));
```

المدخلات السابقة وبعد انشاء الجدول نقوم بادخال

```
insert into stu_study (NO_STU,SUBJECT,MARK) values (111,'216CS',88);

insert into stu_study (NO_STU,SUBJECT,MARK) values (222,'225CS',75);

insert into stu_study (NO_STU,SUBJECT,MARK) values (333,'225CS',40);
```

بعد ذلك نقوم بانشاء الاجراء:

```
declare
mar number(3);
no number(3);
cursor res_stu is
select no_stu,mark
from stu_study;
begin
open res_stu;
loop
fetch res_stu into no,mar;
exit when res_stu%notfound;
if mar>=50 then
update stu_study set result='TRUE' where no_stu=no;
else
update stu_study set result='FALSE' where no_stu=no;
end if;
end loop;
close res_stu;
end;
/
```

وبهذا تكون النتائج في الجدول كمايلي :

NO_STU	SUBJECT	MARK	RESULT
111	216CS	88	TRUE
222	225CS	75	TRUE
333	225CS	40	FALSE

هل رأيتم سهوله ذلك والفائد الكبيرة من المؤشرات. وهناك طريقة اخرى لتعريف المتغيرات لاحظ في الجدول السابق ان الحقل no_Stu تم تعريفه على انه من نوع number وتم تعريف المتغير no في الاجراء على انه number ايضا لكي يتم وضع رقم الطالب فيه لكن لاحظ لو تم تغير نوع الحقل في

الجدول من number الى varchar2 فانه يجب عليك تغير نوع المتغير no في الاجراء ايضا لكن هناك طريقه تجعلك لاتعدل الاجراء كل مرة وهي استخدام الامر التالي لتعريف المتغير no في الاجراء

type%no_stu.stu_study NO

حيث :

NO هي اسم المتغير

stu_study : اسم الجدول

no_stu : الحقل المطلوب في الجدول

type% : خاصية نوع الحقل

ومعنى ماسبق قم بتعريف متغير اسمه no له نفس نوعية الحقل الذي اسمه NO_STU الموجود في الجدول stu_study .

وبهذا لان تقوم بتغير نوع العنصر في الاجراء في كل مرة تغير النوع وهكذا مع جميع المتغيرات التي لها صله بالجداول

وبذلك يصبح الاجراء بعد التعديل كمايلي:

```
declare
mar stu_study.mark%type;
no stu_study.no_stu%type;
cursor res_stu is
select no_stu,mark
from stu_study;
begin
open res_stu;
loop
fetch res_stu into no,mar;
exit when res_stu%notfound;
if mar>=50 then
update stu_study set result='TRUE' where no_stu=no;
else
update stu_study set result='FALSE' where no_stu=no;
end if;
end loop;
close res_stu;
end;
/
```

ب -المؤشرات الضمنية :

وهي اسهل من المؤشرات الصريحه

وتوجد نقطتين هامتين عند التعامل مع المؤشرات الضمنيه :

* يظهر المؤشر الضمني في جسم الاجراء body وليس في declare الخاص بالاجراء كمافي المؤشرات الصريحه

* لابد ان يسترجع مؤشر select الضمني سطر واحد.

والصيغة العامه للمؤشر الضمني كمايلي:

```
SELECT COLUM1,COLUM2,.....  INTO
VARIABLE1,VARIABLE2,.....          FROM  table_name
```

ومعنى هذا قم باختيار الحقل١ و الحقل٢ وضعها في المتغيرات منغير١ و متغير٢ من الجدول table_name

سوف نأخذ مثال على ذلك وسوف نستخدم الجدول الذي انشئناه سابق في الدرس الثاني عندما تعاملنا مع المؤشرات الصريحة وكان اسم الجدول stud

no	name	age
111	mohammed	23
222	talal	22
333	majed	24

واردنا مثلا كتابة اجراء يقوم بحساب متوسط اعمار الطلاب (قد يقول البعض انه لايحتاج ذلك الى اجراء فمجرد استخدام جملة select نستطيع عمل ذلك انا اقول نعم هذا صحيح لكن احب استخدام الاجراء في هذا المثال لكي نرى طريقة عمل المؤشر الضمني ولكن سوف نرى بعد قليل مثال شامل يتم فيه استخدام الموشرات الصريحه والضمنية في نفس الوقت) والان نقوم بكتابة الاجراء كمايلي :

```
set serveroutput on;
declare
aveage number(4,2);
begin
select avg(age)
into aveage
from stud;
DBMS_OUTPUT.PUT_LINE(aveage);
end;
/
```

*** مثال شامل لاستخدام الموشرات الصريحة والضمنيه في نفس الوقت :

لنفرض انه لدينا الجدولين التاليين : الجدول الاول اسمه courses (المقررات):

رقم المقرر	اسم المقرر	عدد الساعات
code	course_name	hours
216CS	NETWORK	3
225CS	ASSEMBLY	3
325CS	DATABASE	4

وقم بانشاء الجدول كمايلي:

```
create table courses(
code varchar2(8),
course_name varchar2(40),
hours number(3),
primary key(code));
```

وقم بادخال البيانات الموجود بالجدول كمايلي:

```
insert into courses values('216CS','NETWORK',3);

insert into courses values('225CS','ASSEMBLY',3);

insert into courses values('325CS','DATABASE',4);
```

ثم نقوم بتكوين الجدول الثاني وهو studys :

اسم الطالب	كود المقرر	الدرجة	عدد النقاط
NO_STU	COURSE_CODE	MARK	POINT
111	216CS	88	
222	225CS	75	
333	225CS	40	
111	225CS	90	
222	216CS	78	
333	216CS	85	

ويتم الانشاء كمايلي:

```
create table studys(
NO_STU varchar2(6),
COURSE_CODE varchar2(8),
MARK number(3),
point number(5,2),
primary key(NO_STU,COURSE_CODE));
```

ويتم ادخال بالبيانات الموجودة بالجدول كمايلي:

```
insert into studys(NO_STU,COURSE_CODE,MARK)
values ('111','216CS',88);
insert into studys(NO_STU,COURSE_CODE,MARK)
values ('222','225CS',75);
insert into studys(NO_STU,COURSE_CODE,MARK)
values ('333','225CS',40);
insert into studys(NO_STU,COURSE_CODE,MARK)
values ('111','225CS',90);
insert into studys(NO_STU,COURSE_CODE,MARK)
values ('222','216CS',75);
insert into studys(NO_STU,COURSE_CODE,MARK)
values ('333','216CS',85);
```

بعد الانتهاء من انشاء وادخال البيانات المطلوب انشاء اجراء يقوم بحساب عدد النقاط لكل طالب وفي كل مادة وهو الحقل عدد النقاط الذي لم ندخل فيه اي شيء ويجب نعلم ان :

MARK	average
95-100	5
90-94	4.75
85-89	4.5
80-84	4
75-79	3.5
70-74	3
65-69	2.5
60-64	2
1-59	1

ويتم حساب النقاط كمايلي :

عدد النقاط في اي مقرر = معدل المادة (وليس الدرجة كمافي الجدول السابق) * عدد ساعات المقرر

مثال لحساب معدل الطالب الذي رقمه ١١١ في المقرر 216CS

نلاحظ من جدول studys ان الطالب قد تحصل على درجة ٨٨ ونلاحظ ان الدرجة من الجدول السابق هي بين ٨٥- ٨٩ وبالتالي فإن معدل الطالب في هذا المقرر هو ٤,٥ (وهي الطريقة المتبعة في اغلب الجامعات) ، ومن جدول courses نحصل على عدد الساعات للمقرر وبالتالي فان :

عدد النقاط = ٤,٥ * ٣ = ١٣,٥ وهكذا في جميع الطلاب وهذا هو المطلوب من الاجراء عمله.

وبالتالي فان الاجراء سوف يكون كمايلي:

```
DECLARE
no_Student studys.NO_STU%type;
hou courses.hours%type;
mark studys.mark%type;
cou_code courses.code%type;
poi studys.point%type;
cursor st_point is
select NO_STU,COURSE_CODE,MARK from studys;
BEGIN
open st_point;
loop
exit when st_point%notfound;
fetch st_point into no_Student,cou_code,mark;
select hours
into hou
from courses
where code=cou_code ;
if (mark>=95)and(mark<=100) then
poi:=5 * hou;
elsif mark>=90 then
poi:=4.75 * hou;
elsif mark>=85 then
poi:=4.5 * hou;
elsif mark>=80 then
poi:=4 * hou;
elsif mark>=75 then
poi:=3.5 * hou;
elsif mark>=70 then
poi:=3 * hou;
elsif mark>=65 then
poi:=2.5 * hou;
elsif mark>=60 then
poi:=2 * hou;
else
poi:=1 * hou;
end if;
update studys set POINT=poi
where NO_STU=no_Student and COURSE_CODE=cou_code ;
end loop;
close st_point;
end;
```

لاحظ هنا اننا استخدمنا المؤشرات الصريحة والمؤشرات الضمنيه والصريحة استخدمناه لكي تقوم بفتح سجلات الجدول studys والمؤشر الضمنلي استخدمنا لكي يعود بعدد الساعات في كل مرة يدور بالحلقة.

شرح الاجراء :

في التعريفات اتوقع انه لاتوجد هناك مشكلة لديكم ، اما جسم البرنامج ابتدأ من begin فهو كمايلي :

اولا يفتح المؤشر الصريح والذي يحتوي على جميع سجلات الجدول studys ثم يكون حلقة دورانية لكي يمر على جميع سجلات الطلاب الموجودة في المؤشر الصريح وطبعا شرط الانهاء لهذه الحلقة هو الوصول الى اخر سجل . ثم يقوم بعملية تحديث سجلات الطالب الاولى في المتغيرات كمايلي:

fetch st_point into no_Student,cou_code,mark;

وطبعا المتغيرات هي رقم الطالب ورقم المقرر والدرجة في المقرر ولنفرض الان نحن الان عند السجل الاول ووهو الطالب الذي رقمه ١١١ ورقم المقرر CS٢١٦ ودرجته هي ٨٨ سوف يضع هذه البيانات في المتغيرات، ثم يستخدم مؤشر ضمني لكي يحضر عدد ساعات المادة التي درسها الطالب ١١١ وهي CS٢١٦ و المؤشر هو

```
select hours
into hou
from courses
where code=cou_code ;
```

ومعنى هذا احضر عدد ساعات المقرر الذي رقمه هو cou_code وهذا المتغير هو معروف من المؤشر الصريح الاول وسبب استخدامنا هذا المؤشر هو ان عدد ساعات المقرر موجودة في جدول اخر ولابد من استخدام هذا المؤشر لكي يحضر عدد الساعات. وبما اننا فرضنا اننا عند السجل الاول فسوف يحضر عدد ساعات المقرر CS٢١٦ وهي ٣ ساعات ثم بدأ يختبر الدرجة وذلك طبقا للجدول الدرجات والمعدلات حيث كانت درجة الطالب الذي رقمه ١١١ في المقرر cCS٢١٦ هي ٨٨ وبالتالي يكون عدد النقاط كمايلي = ٤,٧٥ * ٣ = ١٣,٥ ، وبعد الانتهاء من حساب المعدل يقوم بتعديل الجدول وتحديث قيمة point بقيمتها الجديدة ، وهكذا يمر على كل طالب بنفس الطريقة السابقة الى ان يصل الى نهاية السجلات.وبالتالي تكون النتائج كمايلي في الجدول studys :

NO_STU	COURSE_CODE	MARK	POINT
111	216CS	88	13.5
222	225CS	75	10.5
333	225CS	40	3
111	225CS	90	14.25
222	216CS	78	10.5
333	216CS	85	13.5

بعد الانتهاء من هذا المثال نكون انهينا المؤشرات بنوعيها بشكل تام وبامثله واقعيه وتوصل المعلومة بشكل سليم.

الـــــــــــــجـــــــداول في pl/sql (المصفوفات):

تستخدم هذه الجداول (المصفوفات) مثل المصفوفات في هي لغة من لغات البرمجة مثل لو كانت لديك سلسلة من الارقام وتريد تخزينها فانك تستخدم هذه الجداول للتخزين ويتم تعريف متغير من هذا النوع كمايلي اولا يتم تعريف هذا النوع :

```
TYPE   اسم_النوع IS TABLE OF نوع_المتغير INDEX BY BINARY_INTEGER
```

مثال على ذلك :

```
DECLARE
TYPE num_array IS TABLE OF number(4) INDEX BY
BINARY_INTEGER;
num num_array;
BEGIN
.....
.....
END;
```

لاحظ اولا تم تعريف نوع واسماه num_array ، ثم قام بتعريف متغير num واعطاه نوع num_array وهو النوع الجديد الذي قمنا بانشاءه.مثال عملي /

```
set serveroutput on;
DECLARE
TYPE num_array IS TABLE OF number(4) INDEX BY
BINARY_INTEGER;
i   number(4);
num    num_array;
BEGIN
FOR  i  IN  1..10 LOOP
num(i) := i * i ;
END LOOP;
FOR  i  IN  1..10 LOOP
DBMS_OUTPUT.PUT_LINE(i || '*' || i || '= ' || num(i) );
END LOOP;
END;
/
```

وويكون عمل هذا الاجراء كمايلي : الحلقة الاولى تقوم بضرب العدد i في نفسه وتخزنه في المتغير num برتبه i وهكذا والحلقة الثانية للطباعة ويكون الناتج كمايلي :

1*1= 1
2*2= 4
3*3= 9
4*4= 16
5*5= 25
6*6= 36
7*7= 49
8*8= 64
9*9= 81
10*10= 100

***الاجراءات المخزنة :

شاهدنا في الدروس الماضية ان اي اجراء نقوم بكتابة اني اذا اردت استخدامة اكثر من مرة فاني اقوم بكتابة كل مرة في sql * plus لكي احصل على النتائج لكن ماهو رأيك لو نقوم بتخزين هذا الاجراء في قاعدة البيانات ونعطية اسم وحينما نحتاجه نستدعية باسمه وهذا يوفر علينا الشيء الكثير لذلك درسنا هذا اليوم هو الاجرائيات المخزنة.

ولكي نقوم بانشاء اجراء مخزن نقوم بمايلي :

CREATE [OR REPLACE] PROCEDURE procedure_name(الادخال متغيرات
(الممررة ومتغيرات الاخراج

حيث تمثل procedure_name اسم الاجراء المستخدم.

اما OR REPLACE فهي توضع حينما تعلم ان الاجراء موجود من السابق.

اما عن المغيرات التي بين القوسين فهي اما متغيرات مدخله مثل اذا كان لديك اجراء حساب معدل طالب وتريد تمرير رقم الطالب الذي تريد حساب معدله فهذه هي تعتبر كمدخلات ولتعريف متغير بهذا الشكل يكون كمايلي :

student_id in number(9)

لاحظ اسم المتغير هو student_id ثم بعده وضعنا الكلمة in ومعنى ان هذا المتغير يعتبر كمدخل

اما لتعريف متغير يعود بقيمة من الاجراء مثلا لو اردنا تعرف متغبر يرجع بمعدل الطالب يتم التعريف كمايلي :

ave out number(5,2)

بعد تنفيذ الاجراء يكون هذا المتغير يحتوي على معدل الطالب الذي تم تمرير رقمه مثلا.

مع العلم انه يمكن تعريف متغير للمدخلات والمخرجات حيث تمرر به القيمه اولا وبعد تنفيذ الاجراء يتم وضع القيمه في نفس المتغير وتتم كمايلي :

ave in out number(5,2)

ومعنى هذا اي مدخل ومخرج في نفس الوقت .

مـــــــــــــثال :

في الجدول الذي قمنا بدراسته في الدرس الرابع وكان بأسم studys وكان كمايلي :

NO_STU	COURSE_CODE	MARK	POINT
111	216CS	88	13.5
222	225CS	75	10.5
333	225CS	40	3
111	225CS	90	14.25
222	216CS	78	10.5
333	216CS	85	13.5

لو اردنا تصميم اجراء مخزن لكي يقوم بطباعة درجة الطالب بعد تمرير رقم الطالب ورقم المقرر.

الاجراء المخزن سوف يكون كمايلي :

```
create or replace procedure stu_mark(
stu_id in studys.NO_STU%type,
cou in studys.COURSE_CODE%type)
as
mar studys.mark%type;
begin
select mark
into mar
from studys
where NO_STU=stu_id
and COURSE_CODE=cou;
DBMS_OUTPUT.PUT_LINE(mar);
end;
/
```

بعد الانتهاء من تنفيذ الاجراء يكون الاجراء مخزن في قاعدة البيانات ولكي نقوم باستدعاءه نقوم بمايلي :

```
begin
stu_mark(111,'216CS');
end;
/
```

لاحظ كيف تم استدعاء الاجراء السابق من خلال اسم الاجراء وبذلك سوف يكون الناتج على الشاشة كمايلي 88 وهي صحيحة بعد تمرير رقم الطالب 111 ومقرر 216CS

لكن لاحظ اننا لم نستخدم متغيرات اخراج لكن مأرايك ان نصمم اجراء اخر يقوم بنفس الوظيفة التي يقوم بها الاجراء السابق لكن عملية الطباعة تكون بعد الاستدعاء لكي نجعل الاجراء يقوم بارجاع درجة الطالب بمتغير لذلك فان الاجراء كمايلي :

```
create or replace procedure stu_mark22(
stu_id in studys.NO_STU%type,
cou in studys.COURSE_CODE%type,
mara out studys.mark%type)
as
begin
select mark
into mara
from studys
where NO_STU=stu_id
and COURSE_CODE=cou;
end;
/
```

بعد ذلك نقوم باستدعاء الاجراء ومن ثم طباعة الدرجة لان لو لاحظت الاجراء لايقوم بالطباعة ولاحظ ايضا ان الدرجة تم وضعها في المتغير mara ولذلك سوف يعود بهذه القيمة وسوف يكون الاستدعاء كمايلي :

```
declare
m studys.mark%type;
begin
stu_mark1(111,'225CS',m);
DBMS_OUTPUT.PUT_Line(m);
end;
/
```

وسوف يكون الناتج هو ٩٠ وهذا صحيح بناء على الجدول.

*** الوظائف المخزنة :

تكلمنا في مامضى عن الاجراءات المخزنة واليوم لدينا درس مشابه له وهو الوظائف المخزنة لكن الفرق ان الوظائف لابد ان تعيد قيمة بعد التنفيذ

والصسغة العامة لتكوين وظيفة كمايلي:

CREATE [OR REPLACE] FUNCTION function_name(متغيرات الادخال والاخراج الممررة ومتغيرات)
RETURN datatype

حيث تمثل function_name اسم الوظيفة المستخدمه.

اما REPLACE OR فهي توضع حينما تعلم ان الاجراء موجود من السابق.

اما عن المغيرات التي بين القوسين فهي اما متغيرات مدخله مثل اذا كان لديك اجراء حساب معدل طالب وتريد تمرير رقم الطالب الذي تريد حساب معدله فهذه هي تعتبر كمدخلات ، وهي بنفس الطريقة التي تعاملنا بها مع الاجراءت المخزنة لاتغير على المتغيرات وطرق تعريفها.اما RETURN datatype فهي تدل على نوع القيمة المعادة من الوظيفة .

مثـــــــال : في الجدول الذي قمنا بدراسته في الدرس الرابع وكان بأسم studys وكان كمايلي :

NO_STU	COURSE_CODE	MARK	POINT
111	216CS	88	13.5
222	225CS	75	10.5
333	225CS	40	3
111	225CS	90	14.25
222	216CS	78	10.5
333	216CS	85	13.5

لو اردنا تصميم وظيفة ترجع بمعدل الطالب الفصل اي يتم تمرير رقم الطالب الى الوظيفة ثم يتم حساب المعدل الفصلي للطالب

ويتم حساب المعدل الفصل للطالب كمايلي =مجموع النقاط ÷ مجموع عدد الساعات لمقررات

ولانشاء الوظيفة كمايلي :

```
1  create or replace function stu_avea(stnum in
2 studys.NO_STU%type)
3  return real
4  as
5  hour courses.hours%type;
6  avrage number(4,2);
7  sum_hours courses.hours%type:=0;
8  point studys.POINT%type;
9  total_Point studys.POINT%type:=0;
10 codem courses.CODE%type;
11 cursor sumpoint
12 is
13 select COURSE_CODE,POINT
14 from studys
15 where NO_STU=stnum;
16 begin
17 open sumpoint;
18 loop
19 fetch sumpoint into codem,point;
20 exit when sumpoint%notfound;
21 select hours
22 into hour
23 from courses
24 where code=codem;
25 total_Point:=total_Point+point;
26 sum_hours:=sum_hours+hour;
27 end loop;
28 close sumpoint;
29 avrage:=total_Point/sum_hours;
30 return avrage;
   end;
```


الشرح :

السطر رقم ١: لتعريف الوظيفة
السطر رقم ٢: نوع القيمة التي سوف ترجع بها الوظيفة
السطر رقم ٤: تعريف متغير عدد الساعات وهو نفس حقل عدد ساعات المقرر الموجودة في جدول courses
السطر رقم ٥: تعريف متغير الذي سوف نضع به المعدل

السطر رقم ٦: تعريف متغير لكي يوضع به مجموعات الساعات للطلاب في كل المواد
السطر رقم ٧: تعريف متغير لكي يوضع به عدد نقاط الطلاب في اي مقرر
السطر رقم ٨: تعريف متغير لكي يوضع به مجموع عدد نقاط الطالب في كل المقرر
السطر رقم ٩: تعريف متغير لكود المادة
السطر رقم ١٠: تعريف مؤشر صريح للحصول على كود المادة لكي نستفيد منه في الحصول على عدد الساعات وعدد النقاط في ذلك المقرر لكي نضيفها الى مجموع النقاط
السطر رقم ١٦: فتح هذا المؤشر لكي نتعامل معه
السطر رقم ١٧: الدخول على حلقة لكي نمر على جميع الجدول
السطر رقم ١٨: تحديث قيم المؤشر للسجل الحالي في المتغيرات codem,point
السطر رقم ١٩: شرط انهاء الحلقة وهو اذا لم يجد اي سجل في المؤشر
السطر رقم ٢٠: مؤشر ضمني لكي يقوم بالحصول على عدد ساعات الطلاب في المقرر الموجود حاليا في المؤشر الصريح ويضع عدد الساعات في المتغير hour
السطر رقم ٢٤: اضافة عدد النقاط للمقرر الحالي الى مجموع النقاط السابق
السطر رقم ٢٥: اضافة عدد الساعات للمقرر الحالي الى مجموع الساعات السابق
السطر رقم ٢٦: الخروج من الحلقة
السطر رقم ٢٧: انهاء المؤشر الضمني
السطر رقم ٢٨: حساب المعدل وهو مجموع النقاط تقسيم مجموع عدد الساعات
السطر رقم ٢٩: الرجوع بقيمة المعدل
السطر رقم ٣٠: الانهاء

الان بعد الانتهاء من شرح طريقة تصميم الوظيفة جاء دور طريقة الاستدعاء :

لكن قبل الاستدعاء لنحسب يدويا معدل الطالب الذي رقمه ١١١ مثل لكي نقارنه بالنتائج بعد الاستعلام :

مجموع نقاط الطالب = ١٣,٥+١٤,٢٥= ٢٧,٧٥

مجموع عدد الساعات = (عدد ساعات المقرر CS٢١٦) + (عدد ساعات المقرر CS٢٢٥)
= ٣ + ٣ = ٦

وبالتالي فان معدل الطالب = ٢٧,٧٥ ÷ ٦ = ٤,٦٣

لكن الان دعنا نستدعي الدالة ونشاهد النتائج

```
SELECT distinct(NO_STU),stu_avea(no_stu)
from studys
where no_stu=111;
```

لاحظ كيف تم استدعاء الدالة من خلال الاستعلام ولاحظ استخدام الدالة distinct وهي لعدم تكرار السجل واليك النتائج :

```
NO_STU              STU_AVEA(NO_STU)
------      ;       ----------------
111                       4.63
```

لاحظ لو كان الاستعلام بدون وجود الدالة distinct فسوف يتكرر رقم الطالب عدد ظهوره في الجدول لذلك لوكان كمايلي :

```
SELECT NO_STU,stu_avea(no_stu)
from studys
where no_stu=111;
```

فان النتائج ستصبح هكذا

```
NO_STU              STU_AVEA(NO_STU)
------      ;       ----------------
111                       4.63
111                       4.63
```

وهذا سبب ظهور الدالة distinct

** الحزم package :

تعلمنا سابق كيفية انشاء الاجراءت والوظائف المخزنة. لكن مارأيك لو وجد لدينا قاعدة بيانات كبيرة جدا ولنضرب مثال انها تحتوي على ٥٠ اجراء او وظيفة وظيفة او اجراء لها عمل خاص ولنفرض ان هذه القاعدة هي لمحل تجاري ضخم يحتوي على بيانات العملاء وبيانات الموظفين وبيانات الاصناف التجارية وبيانات المخزون وغيرها من بيانات ، ولذلك فان بعض هذه الاجرائيات والوظائف المخزنة مختص بالعملاء مثلا وجود اجراء لحساب اجمالي عميل وغيرها من الاجرائيات ، ومثل وجود اجرئيات خاصة بالموظفين مثلا اجرائية خاصة بحساب راتب الموظف بعد حذف الحسومات واضافة العلاوات وغيرها ايضا ، لكن وضعها في هذا الشكل في قاعدة البيانات قد يسبب لك بعض الارباك لذلك مارأيك بان تجمع كل الوظائف والاجرائيات الخاصة بكل قسم في مجموعة لوحدها وهذه المجموعة تدعي الحزمه package مثلا نجمع كل اجرائيات والوظائف الخاصة بالعملاء في حزمة خاصة

فوائد استخدام الحزمة:

١- تجميع وحدات pl/sql المرتبطة.
٢- اداء افضل.
٣- تكون السرية افضل.
٤- اهم شيء هو في عملية الصيانه حيث تسهل عملية الصيانة باستخدام الحزم.

مكونات الحزم:

تتكون الحزمة من جزءين الاول وهو الوصف specification ويحتوي على التعاريف مثل متغيرات او مؤشرات او اسماء الاجراءت ومتحولتها .
اما الجز الثاني فهو جسم الحزمة ويحتوي على تفاصيل الاجراءت والعمليات وغيرها
والصيغة العامة لانشاء الجزء الاول كمايلي:

```
CREATE OR REPLACE PACKAGE pack_name AS
.....
.....
.....
end;
```

والصيغة العامة لانشاء الجزء الثاني كمايلي :

```
CREATE OR REPLACE PACKAGE BODY pack_name AS
.....
جسم الحزمة
.....
end;
```

لكن يجب ان يكون اسم الحزمة في الجزء الاول هو نفس اسم الحزمة في الجزء الثاني.

مـــــــــــــثال:

لنقم بانشاء حزمة تحتوي على وظيفة لحساب معدل طالب واجراء لطباعة المعدل
ولذلك سوف نستخدم نفس الوظيفة التي انشناها في الدرس السادس والتي اسمها stu_avea والتي تقوم بحساب معدل الطالب والان نبداء بانشاء الحزمة . الجزء الاول من الحزمة specification كمايلي:

```
CREATE OR REPLACE PACKAGE student AS
function stu_avea(stnum in studys.NO_STU%type)return real;
procedure print_ave(avrage in real);
end;
```

الان نقوم بانشاء جسم الحزمة والتي تحتوي على التفاصيل.كمايلي

```
CREATE OR REPLACE PACKAGE BODY student AS
function stu_avea(stnum in studys.NO_STU%type)
return real
as
hour courses.hours%type;
avrage number(4,2);
sum_hours courses.hours%type:=0;
point studys.POINT%type;
total_Point studys.POINT%type:=0;
codem courses.CODE%type;
cursor sumpoint
is
select COURSE_CODE,POINT
from studys
where NO_STU=stnum;
begin
open sumpoint;
loop
fetch sumpoint into codem,point;
exit when sumpoint%notfound;
select hours
into hour
from courses
where code=codem;
```

```
total_Point:=total_Point+point;
sum_hours:=sum_hours+hour;
end loop;
close sumpoint;
avrage:=total_Point/sum_hours;
return avrage;
end;

procedure print_ave(avrage in real)
as
begin
DBMS_OUTPUT.PUT_LINE(avrage);
end;
end;
```

ويحتوي جسم الحزمة كمانلاحظ على مكونات الوظيفة والاجراء الذي تم تعريفهما في وصف الحزمة حيث ان الوظيفة لحساب المعدل والاجراء لطباعة المعدل.

*** طريقة استدعاء اجراء او وظيفة موجود داخل حزمة :

تتم عملية الاستدعاء كمايلي : pack_name.func_proc_name

اي اسم الحزمة اولا ثم نقطة ثم اسم الاجراء او الوظيفة مـــــثال :

```
set serveroutput on
declare
aa real;
begin
aa:=student.stu_avea(111);
student.print_ave(aa);
end;
/
```

وبعد التفيذ يكون الناتج هو معدل الطالب الذي رقمه ١١١ لاحظ اول شي استدعينا داله حساب المعدل ووضعناها في المتغير aa ثم استدعينا اجراء الطباعة ليتم طبعة على الشاشة.

مارأيك من الان فصاعد ان تستخدم الحزم في كتابة والان و بعد ان تعرفت على فائدة الحزم الاجرائيات والوظائف

-- التعامل مع الاجرائيات والوظائف المخزنة :

يوفر النموذج OO40 الوظائف اللازمة للتعامل مع الاجراءات والوظائف المخزنة في قاعدة البيانات

* استدعاء اجراء مخزن بسيط :

استدعاء اجراء مخزن من خلال النموذج OO40 لايختلف كثيرا عن استدعائه من خلال * sql plus .

دعنا ننشئ جدولا لاستخدامه في مثال بسيط:

```
Create table massege(
Msg varchar2(50));
```

ثم نقوم بانشاء اجراء لكي يقوم بادخال البيانات الى الجدول كمايلي :

```
Create or replace procedure ins_messge(mesg in varchar2) is
Begin
Insert into  massege values(mesg);
Commit;
End ;
/
```

ولعمل هذا الاجراء من خلال sql *plus نقوم بمايلي :

```
begin
ins_messge('mohammed');
end;
/
```

وفيمايلي تنفيذ نفس الاجراء من خلال النموذج OO40 :

الكود :

```
Dim ses As OraSession

Dim dat As OraDatabase

Dim dyna As OraDynaset

Set ses = CreateObject("oracleinprocserver.xorasession")

Set dat = ses.OpenDatabase("mohammed", "scott/tiger", 0&)

dat.ExecuteSQL ("begin ins_messge('mohammed'); end;")

MsgBox "تم ادخال البيانات"
```