

الأكاديمية العربية الدولية



الأكاديمية العربية الدولية
Arab International Academy

الأكاديمية العربية الدولية المقررات الجامعية

مقدمة قواعد البيانات

فهرس الكتاب

الصفحة	الموضوع	م
4	الفصل الأول: أساسيات لغة الاستفسارات الهيكلية	
5	• مقدمة	1-1
10	• لغة الاستفسارات الهيكلية SQL	2-1
13	• استخدام SQL*PLUS	3-1
15	• جملة SELECT	4-1
21	• جملة الشرط Where	5-1
36	• ترتيب النتائج باستخدام order by	6-1
40	• القيمة null وتأثيراتها	7-1
42	• استخدام محرر النصوص في SQL*PLUS	8-1
43	• الأسئلة والتمارين	9-1
50	الفصل الثاني: تقنيات لغة SQL	
53	• الرابط بين جدولين أو أكثر	1-2
65	• استخدام Multi-row functions	2-2
71	• استخدام single-row functions	3-2
82	• استخدام group by clause	4-2
86	• استخدام having clause	5-2
89	• استخدام و عمل إستعلام فرعى sub query	6-2
99	• الأسئلة والتمارين	7-2
105	الفصل الثالث: معالجة البيانات باستخدام SQL	
108	مقدمة	1-3
110	• إضافة سجلات جديدة للجداول	2-3
113	• حذف بيانات موجودة في قاعدة البيانات	3-3
115	• تعديل بيانات موجودة في قاعدة البيانات	4-3
121	• إنشاء الجداول وإضافة المحددات لها	5-3
136	• الأسئلة	6-3
143	الفصل الرابع :ورشة عمل	
147	التدريب العملي	1-4

الصفحة	الموضوع	م
150	الفصل الخامس :مصمم النماذج	
154	مقدمة عن oracle developer	1-5
155	أدوات البناء في oracle developer	2-5
159	المكونات الأساسية لمنشيء النماذج	3-5
165	إنشاء النماذج باستخدام المعالج	4-5
175	تنفيذ تطبيق منشيء النماذج	5-5
180	خصائص الكائنات الموجودة على النموذج	6-5
192	الأسئلة	7-5

الفصل الأول

أساسيات لغة الاستفسارات الهيكالية

SQL

الأهداف:

إعطاء الطالب صورة موضحة عن أساسيات لغة الاستعلامات الهيكلية SQL وكيفية الدخول إلى برنامج SQL*PLUS وإدخال تعليمات SQL وكيفية كتابة جملة SELECT واستخدام كلا SELECT مع جملة WHERE، ORDER BY من

المحتويات:

1- مقدمة

2- الولوج إلى برنامج SQL*PLUS

3- إدخال تعليمات SQL

4- استخدام جملة Select ، تعریف Alias

5- جملة الشرط Where

6- ترتيب النتائج باستخدام Order by Clause

7- مفهوم القيمة Null وتأثيراتها

8- استخدام محرر النصوص في برنامج SQL*PLUS

ماذا سنتعلم في هذا الفصل:

في نهاية هذا الفصل يكون الطالب قد اكتسب المهارات والمعرفات التالية:

(1) التفرقة بين قواعد البيانات ونظم إدارة قواعد البيانات

(2) التعرف على لغة الاستعلامات الهيكلية (SQL)

(3) التفرقة بين SQL*Plus ، SQL Statement

(4) استخدام جملة Select ، تعريف Alias

(5) استخدام جملة الشرط Where

(6) ترتيب النتائج باستخدام Order by

(7) استخدام القيمة NULL في التعبيرات الحسابية

(8) استخدام محرر النصوص في برنامج SQL*PLUS

مقدمة

برامج قواعد البيانات من أوسع برامج الحاسوب انتشارا ويمكن أن تستخدم سواء في المجالات التجارية أو الصناعية ومنها على سبيل المثال مجالات النظم المحاسبية والمالية ومجال شؤون الأفراد والمشتريات، وتعتبر لغة الاستعلامات الهيكلية من أهم وأشهر اللغات المستخدمة في برمجة قواعد البيانات من حيث إنشاء ملفات قواعد البيانات واسترجاع البيانات منها بطرق مختلفة ومتنوعة تلائم معظم احتياجات المبرمجين وتتميز بسهولة تركيبها وسهولة الفهم للجميع.

وتستخدم لغة الاستعلام مع الكثير من التطبيقات حيث صممت في الأصل لاستخدام لغات البرمجة الأخرى، ولقد استطاعت SQL أن تحقق طموحات معظم مطوري أنظمة قواعد البيانات حيث أنها لغة قياسية وفعالة في نفس الوقت لبناء ومعالجة قواعد البيانات، ولهذا السبب فإن أي شخص يريد الدخول عالم قواعد البيانات أن يتعلم أولاً SQL، ويعتبر نظام أوراكل من أفضل نظم قواعد البيانات العلاقية وهو يتميز بإمكانية إدارة قواعد بيانات كبيرة جداً من المعلومات والتي قد تصل حتى تيرابايت من المعلومات وإمكانية التعامل مع عدد كبير جداً من المستخدمين بشكل متزامن بالإضافة إلى ميزات الأمان والوثوقية العالية جداً، وظهرت عدة إصدارات من Oracle Developer وأخرها هو Oracle 9i وهو البيئة التي سنقوم بدراسة لغة الاستعلامات الهيكلية من خلالها.

١- قواعد البيانات

١- المقصود بقاعدة البيانات

هي مجموعة مترابطة من البيانات يمكن استخدامها لاستخراج المعلومات التي تريدها أو هي عبارة عن تجمع لكمية كبيرة جداً من البيانات والمعلومات وعرضها بطريقة تسهل من الاستفادة منها فعلي سبيل المثال فإن دليل الهاتف هو أبسط قاعدة بيانات يمكن أن تتعامل معها فيمكنك تسجيل أسماء أصدقائك وأرقام telephones الخاصة بهم حتى يمكنك فيما بعد أن تبحث عن اسم أحد الأصدقاء بإدخال رقم الإشراك أو العنوانوهكذا.

٢- نظم إدارة قواعد البيانات

نظم إدارة قواعد البيانات (DBMS) هي مجموعة من البرامج الجاهزة التي تقوم بتنفيذ جميع الوظائف والمهام المطلوبة من قاعدة البيانات فعلى سبيل المثال بعد إدخال بيانات العاملين إلى قاعدة بيانات شؤون العاملين فإنه قد تحتاج إلى ترتيب أسماء العاملين أبجدياً فإن مثل هذا العمل يطلق عليه إدارة قاعدة البيانات، ويكون نظام إدارة قواعد البيانات من مجموعة من البرامج والملفات التي تتشابك مع بعضها لحل مشكلة أو تحويل نظام يدوي إلى نظام يعمل بالحاسوب مثل تحويل حسابات المخازن من أنظمة ودفاتر يومية إلى نظام وملفات تستخدم بواسطة الحاسوب الآلي.

٣- أنواع قواعد البيانات

• قواعد بيانات هرمية (Hierarchy Database)

• قواعد بيانات شبكية (Network Database)

• قواعد بيانات علائقية (Relational Database)

يقتصر استخدام كلاً من قواعد البيانات الهرمية والشبكية على الحاسوبات الكبيرة، أما قواعد البيانات العلائقية Relational Database فإنها أكثر استخداماً وشهرة مع الحاسوب وإليها ينتمي Oracle For Windows.

2- لغة الاستفسارات الهيكيلية (SQL)

تمكنك لغة SQL (اختصاراً للكلمات Structured Query Language) من إدارة قواعد البيانات بشكل كامل وإجراء جميع العمليات القياسية كإنشاء الجداول وتعديلها بالبيانات، أو إجراء الاستعلامات عليها وكذلك الرابط بين الجداول المختلفة، ولكن كيف يمكننا عمل ذلك بها؟

كما وضمنا سابقاً فإن قاعدة البيانات عبارة عن مكان أو مستودع كبير لتخزين البيانات المختلفة، تريحك قاعدة البيانات من عناية تخزين بياناتك في ملفات منفصلة وكما أنه تحتاج إلى إجراء عمليات البحث والتصنيف لهذه البيانات عن طريق خوارزميات البحث المعقدة فتسهل لك ذلك بأنها تعطيك واجهة سهلة للتعامل مع البيانات، بحيث لا تحتاج إلى كل هذا وتكون البيانات في قاعدة البيانات مخزنة في عدة جداول Tables وكما نعلم فالجدول يتكون من صفوف Rows وأعمدة Columns هكذا:

الجدول السابق يتكون من ثلاثة صفوف وثلاثة أعمدة، وفي قواعد البيانات فإننا نسمي الصفوف بالسجلات Records ونسمى الأعمدة بالحقول Fields، مما هي الحقول وما هي السجلات؟

يقوم الحقل الواحد في الجدول بتخزين معلومة معينة، فمثلاً عندما يكون لدينا قاعدة بيانات بأرقام الهواتف فإننا سنحتاج إلى حقولين (أو عمودين) واحد لاسم والثاني لرقم الهاتف، أما السجلات (أو الصفوف) فيحتوي كل منها على مجموعة من الحقول، فترى بأن السجل الواحد في مثالنا يحتوي على معلوماتين مختلفتين هما الاسم ورقم الهاتف، لذلك يمكننا القول بأن قاعدة البيانات تتكون من الجداول والجداول تتكون من السجلات والسجلات تتكون من الحقول، وكل حقل يحتوي على معلومة واحدة، أي أن الحقل هو أصغر وحدة قاعدة البيانات.

SQL-1 عبارات

تنقسم عبارات SQL إلى ثلاثة فئات رئيسية وهي كالتالي:

DDL-1-1 لغة تعريف البيانات

هي مجموعة من أوامر SQL تستطيع من خلالها إنشاء وتعريف الكائنات في قاعدة البيانات، وتقوم هذه الأوامر بإنشاء أو إسقاط أو تغيير كائن قاعدة البيانات، وهي اختصاراً Create, Alter, Drop, Rename, Truncate (Data Definition Language) للكلمات

2-1-2 لغة معالجة البيانات DML

هي مجموعة من العبارات التي تستطيع من خلالها معالجة البيانات الموجودة في قاعدة البيانات من إدراج البيانات وتحديثها أو تحديد أو حذف البيانات، وهي اختصارا Insert, (Data Manipulation Language)، ومن أهم أوامرها Update, Delete للكلمات

2-1-3 لغة التحكم والصلاحيات DCI

هي مجموعة من أوامر SQL تستطيع من خلالها منح أو سحب إمتيازات استخدام كائن قاعدة البيانات، وهي اختصارا للكلمات، وهي اختصارا للكلمات (Data Control Language)، ومن أهم أوامرها Grant, Revoke

وإيجازاً للقول فإن الشكل التالي يوضح الفئات الرئيسية لعبارات SQL

INSERT	لغة معالجة البيانات
UPDATE	
DELETE	
CREATE	
ALTER	
DR	
OP	لغة تعريف البيانات
RENAME	
TRUNCATE	
GRANT	
REVOKE	لغة التحكم والصلاحيات
COMMIT	
ROLLBACK	لغة معالجة البيانات
SAVEPOINT	Transaction control

SQL*Plus -3

يعتبر برنامج SQL*PLUS أحد منتجات شركة Oracle، وهو واجهة التخاطب مع قاعدة البيانات، ويستخدم في إنشاء قواعد البيانات ومنح السماحيات، وإنشاء الجداول وتعديلها واسترجاع البيانات من الجداول.

SQL*Plus استخدام 1-3

يعد استخدام SQL*Plus من أسهل الطرق لتعلم لغة SQL، للدخول على SQL*Plus يتم ذلك بالطريقة التالية:

✓ من قائمة Programs ، ثم start أشر إلى Oracle -

Development Application، منه OraHome9i SQL*Plus

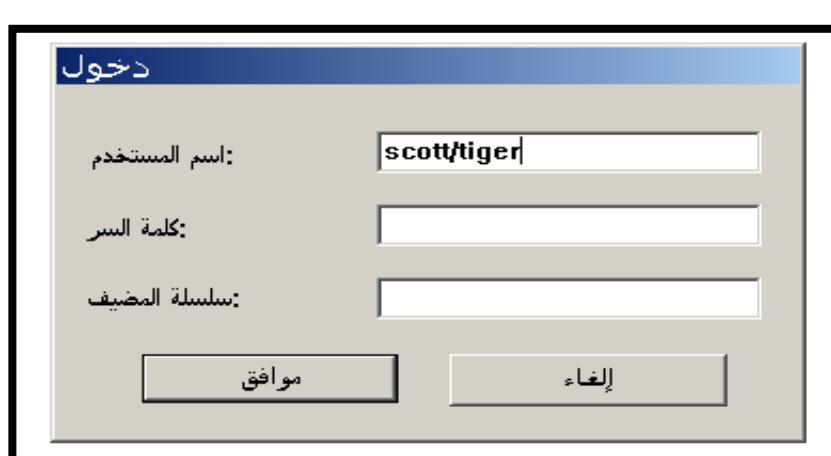
ستظهر لك الشاشة التالية تسأل عن اسم المستخدم وكلمة المرور

• اسم المستخدم (user name): إسم معرف حساب مستخدم oracle الذي تم تحديده للمستخدم.

• كلمة المرور (password): كلمة المرور الخاصة بحساب المستخدم.

• سلسلة المصنف (Host): سلسلة الأحرف التي تحدد مكان تواجد قاعدة البيانات، وتسمى بسلسلة الاتصال (connect string) وتستخدم عندما تكون متصلًا بشبكة oracle الذي تريد الوصول إليه قد تم إعداده على جهاز آخر

غير الحاسب الذي تقوم بالتعامل منه مع oracle



Select - جملة

الأمر Select يمكنك من استعادة البيانات من داخل قاعدة البيانات، وتأخذ جملة

الشكل التالي Select

```
SELECT [DISTINCT] {*, column  
[alias], ...}
```

1- اختيار كافة الأعمدة

تمثل عبارة * طريقة مختصرة لإخبار SQL بأننا نريد إظهار جميع

أعمدة الجدول، فعلى سبيل المثال

```
SQL> SELECT *  
2 FROM emp;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17/12/80	800		20
7499	ALLEN	SALESMAN	7698	20/02/81	1600	300	30
7521	WARD	SALESMAN	7698	22/02/81	1250	500	30
7566	JONES	MANAGER	7839	02/04/81	2975		20
7654	MARTIN	SALESMAN	7698	28/09/81	1250	1400	30
7698	BLAKE	MANAGER	7839	01/05/81	2850		30
7782	CLARK	MANAGER	7839	09/06/81	2450		10
7788	SCOTT	ANALYST	7566	19/04/87	3000		20
7839	KING	PRESIDENT		17/11/81	5000		10
7844	TURNER	SALESMAN	7698	08/09/81	1500	0	30
7876	ADAMS	CLERK	7788	23/05/87	1100		20
7900	JAMES	CLERK	7698	03/12/81	950		30
7902	FORD	ANALYST	7566	03/12/81	3000		20
7934	MILLER	CLERK	7782	23/01/82	1300		10

تم اختيار 14 صف

```
SQL>
```

Select (1) : تحدد ما هي الأعمدة المراد إظهارها.

From (2) : تحدد المكان الذي نريد استعادة البيانات منه (جدول أو أكثر)

(3) الفاصلة المنقوطة: تخبر SQL بأن العبارة كاملة وجاهزة للتنفيذ.

(4) الأقواس [] تشير على أن ما يدخل هذه الأقواس اختيارياً.

2 اختيار أعمدة محددة

يمكننا اختيار أعمدة محددة، وهو أمر في غاية السهولة حيث تقوم بكتابة أسماء الأعمدة بعد

عبارة Select، مع وضع فاصلة بين أسماء الأعمدة، ثم تحدد الجدول بعد From

SQL> SELECT empno, ename, job
2 FROM emp;

مثال 2

EMPNO	ENAME	JOB
7369	SMITH	CLERK
7499	ALLEN	SALESMAN
7521	WARD	SALESMAN
7566	JONES	MANAGER
7654	MARTIN	SALESMAN
7698	BLAKE	MANAGER
7782	CLARK	MANAGER
7788	SCOTT	ANALYST
7839	KING	PRESIDENT
7844	TURNER	SALESMAN
7876	ADAMS	CLERK
7900	JAMES	CLERK
7902	FORD	ANALYST
7934	MILLER	CLERK

تم اختيار 14 صف

✓ لا تنس إنتهاء جملة Select بالفاصلة المنقوطة لكي تخبر SQL بأن الجملة

كاملة وجاهزة للتنفيذ.

✓ تظهر الأعمدة المطلوبة بدءاً من اليسار إلى اليمين بنفس ترتيب كتابتها في عبارة

Select

مثال 3

```
SQL> SELECT deptno, dname, loc
  2 FROM      dept;
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

4-3 التعبيرات الحسابية (Arithmetic Expressions)

- ضع التعبيرات الحسابية بعد عبارة Select كأي عمود تريد استعادة بيانته، حيث

تستخدم التعبيرات الحسابية مع تلك الأعمدة التي نوع معطياتها عددي مثل الأرقام

وال تاريخ.

- يجب عليك الانتباه لأسبقية المعاملات (Operator Precedence) في

التعبيرات الحسابية وفيما يلي بيان بأسبقية المعاملات في التعبيرات الحسابية

1) ينفذ أولاً الضرب أو القسمة أولاً، ومن اليسار إلى اليمين .

2) بعد ذلك ينفذ الجمع أو الطرح لاحقاً، ومن اليسار إلى ومن اليسار إلى اليمين .

تلخيص

✓ عند استخدام التعبيرات الحسابية يمكنك تغيير تسلسل أسبقية المعاملات الحسابية

باستخدام الأقواس (Parentheses) ، حيث تأخذ الأقواس الأسبقية في التعبيرات

الحسابية .

4-3-1 أمثلة علي التعبيرات الحسابية

SQL> SELECT ename, sal, sal+500
2 FROM emp;

مثال 1

في هذا المثال يقوم باستخدام معامل الجمع لزيادة مرتبات جميع الموظفين ب 500 ريال،
ويقوم بعرض الناتج في عمود جديد .

ENAME	SAL	SAL+500
SMITH	800	1300
ALLEN	1600	2100
WARD	1250	1750
JONES	2975	3475
MARTIN	1250	1750
BLAKE	2850	3350
CLARK	2450	2950
SCOTT	3000	3500
KING	5000	5500
TURNER	1500	2000
ADAMS	1100	1600
JAMES	950	1450
FORD	3000	3500
MILLER	1300	1800

14 row selected

تلميح

→ تتجاهل SQL*Plus المسافات Blank Spaces قبل وبعد

SQL> SELECT ename, sal, 12*sal+500
2 FROM emp;

تلميح

المعاملات الحسابية .

في هذا المثال سيقوم طبقاً لأسبقية المعاملات كما وضمنا سابقاً بحساب عملية الضرب أولاً

وهي $12 * \text{sal}$ ، ثم إجراء عملية الجمع ثانياً وهي إضافة 500 ريال

ENAME	SAL	$12 * SAL + 500$
SMITH	800	10100
ALLEN	1600	19700
WARD	1250	15500
JONES	2975	36200
MARTIN	1250	15500
BLAKE	2850	34700
CLARK	2450	29900
SCOTT	3000	36500
KING	5000	60500
TURNER	1500	18500
ADAMS	1100	13700
JAMES	950	11900
FORD	3000	36500
MILLER	1300	16100

تم اختيار 14 صف

SQL>

✓ في المثال السابق كنا نريد زيادة مرتبات جميع الموظفين ب 500 ريال ثم حساب

المرتب السنوي بعد عملية الزيادة فكيف يمكننا ذلك . . .؟

✓ كما تعلمنا سابقاً يمكنك تغيير تسلسل أسبقية المعاملات الحسابية باستخدام

الأقواس (Parentheses) ، حيث تأخذ الأقواس الأسبقية في التعبيرات الحسابية،

ويتم ذلك على النحو التالي

تمرين

تمرين

SQL> SELECT ename, sal, 12*(sal+500)
2 FROM emp;

تعريف-5

ENAME	SAL	12*(SAL+500)
SMITH	800	15600
ALLEN	1600	25200
WARD	1250	21000
JONES	2975	41700
MARTIN	1250	21000
BLAKE	2850	40200
CLARK	2450	35400
SCOTT	3000	42000
KING	5000	66000
TURNER	1500	24000
ADAMS	1100	19200
JAMES	950	17400
FORD	3000	42000
MILLER	1300	21600

تم اختيار 14 صف

SQL> |

يمكننا إعادة تسمية رؤوس الأعمدة باستخدام column alias a، بمعنى إبدال

الاسم الفعلي للحقل باسم اعتباري يتم تحديده بواسطة المستخدم ويكون ذلك مفيداً وخاصة

عند التعامل مع التعبيرات الحسابية، وفيما يلي عدة اعتبارات لابد من مراعاتها عند إعادة تسمية

رؤوس الأعمدة

(1) يتم كتابة alias في جملة Select يلي اسم العمود أو التعبير الحسابي المراد إعادة

تسميتها مع ترك مسافة .

(2) يتم استخدام as بين اسم العمود وال alias، ويكون ذلك اختياريا .

(3) إذا احتوى alias على مسافات أو أحرف وعلامات خاصة مثل (# or \$)

من وضعة بين علامتي تصيص (" double quotation(")

مثال 1

مثال 1

```
SQL> SELECT ename, sal AS Salary
2 FROM emp;
```

ENAME	SALARY
SMITH	800
ALLEN	1600
WARD	1250
JONES	2975
MARTIN	1250
BLAKE	2850
CLARK	2450
SCOTT	3000
KING	5000
TURNER	1500
ADAMS	1100
JAMES	950
FORD	3000
MILLER	1300

تم اختيار 14 صف

SQL>

للمزيد

✓ في المثال السابق تم إعادة تسمية عمود Sal في جدول emp بـ salary

✓ تم وضع كلمة AS قبل alias name، وكما تعلمنا أن ذلك اختيارياً.

مثال 2

✓ يتم ظهور alias name بحالة أحرف كبيرة uppercase

```
SQL> SELECT ename, sal*12 "annual salary"
2 FROM emp;
```

ENAME	Annual Salary
SMITH	9600
ALLEN	19200
WARD	15000
JONES	35700
MARTIN	15000
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200
JAMES	11400
FORD	36000
MILLER	15600

تم اختيار 14 صف

في هذا المثال تم إعادة تسمية التعبير الحسابي بالـ $sal * 12$

كما نلاحظ تم وضع "Annual Salary" بين علامتي تصيص .Spaces لأنة يحتوي مسافات

6- جملة الشرط Where Clause

كان اختيارنا سابقاً للأعمدة يجري علي كافة الصفوف التي يحتويها الجدول، أردنا إظهار صفوفاً تحتوي على قيمة معينة، كإظهار الموظفين الذين تتجاوز مرتباتهم 3000 ريال أو الذين يقطنون مدينة ما، لفعل مثل ذلك سنحتاج إلي وضع كلمة Where ضمن عبارة Select، يأخذ الاختيار الشرطي الشكل التالي

```
SELECT      [DISTINCT] {*, column [alias], ...}
  FROM        table
 [WHERE       condition(s)];
```

✓ تأتي جملة Where Clause بعد جملة From Clause

للمزيد

✓ يلي Where يبي شرط أوعدة شروط

✓ توجه عبارة Where الأمر إلي أوراكل بالبحث في قاعدة البيانات واستعادة

الصفوف التي تتواافق مع شرط البحث.

مثال 1

```
SQL> SELECT *
  2  FROM    emp
  3 WHERE   ename='tarek';
```

في هذا المثال نريد استرجاع جميع الأعمدة بجدول emp، ولكن بشرط أن يكون

اسم الموظف tarek

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	tarek	MANAGER	7839	01/01/99	5000	3000	30

مثال 2

```
SQL> SELECT *
2 FROM emp
3 WHERE hiredate='23-JAN-82';
```

في هذا المثال نريد استرجاع جميع الأعمدة بجدول emp، ولكن بشرط أن يكون تاريخ
تعيين الموظف هو '23-JAN-82'

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	tarek	MANAGER	7839	01/01/99	5000	3000	30

✓ الأحرف النصية Character Strings، والتاريخ في جملة Where

لابد من وضعها بين علامتى تصييص (' ')

quotation

مثال 3

```
SQL> SELECT ename, job, deptno
  FROM emp
 WHERE ename = 'MILLER';
```

1- استخدام عوامل المقارنة (Comparison Operators)

تحمّل SQL ستة عوامل للمقارنة نستطيع استخدامها مع Where، كما هو مبين بالجدول

التالي

Operator المعامل	Meaning المعنى
=	يساوي
>or <	أكبر من أو أقل من
>=	أكبر من أو يساوي
<=	أقل من أو يساوي
<> Or ! =	لا يساوي

```
SQL> SELECT ename, job, sal  
2 FROM emp  
3 WHERE sal >= 3000;
```

مثال 1

في هذا قد المثال استخدمنا أحد عوامل المقارنة

ENAME	JOB	SAL
SCOTT	ANALYST	3000
KING	PRESIDENT	5000
FORD	ANALYST	3000

في جملة Where، حيث يتم استعادة

عندما يكون ename، job، sal أكيرمن أو يساوى 3000 ريال.

مثال 2

```
SQL> SELECT ename, job, sal  
  2 FROM emp  
  3 WHERE sal <> 3000;
```

ENAME	JOB	SAL
SMITH	CLERK	800
ALLEN	SALESMAN	1600
WARD	SALESMAN	1250
JONES	MANAGER	2975
MARTIN	SALESMAN	1250
BLAKE	MANAGER	2850
CLARK	MANAGER	2450
KING	PRESIDENT	5000
TURNER	SALESMAN	1500
ADAMS	CLERK	1100
JAMES	CLERK	950
MILLER	CLERK	1300

12 row selected

6-2 استخدام عوامل المقارنة الأخرى

(Other Comparison Operator)

تتضمن SQL أربعة معاملات أخرى بالإضافة للمعاملات التي الإشارة إليها في النقطة

السابقة يمكن استخدامها مع جملة Where، كما هو مبين بالجدول التالي

المعامل Operator	المعنى Meaning
BETWEEN.....AND.....	بين قيمتين
IN (list)	تتيح لك اختيار أي قيمة من List
LIKE	يشابه أو يماثل
IS NULL	فارغاً

6-1-2 Between استخدام معامل

يمكنك إجراء عمليات البحث داخل نطاق معين من القيم وذلك باستخدام معامل Between مع جملة Where، فعلى سبيل المثل قد ترغب في استعادة بيانات جميع الموظفين الذين يتلقاون راتب مابين 2000، 5000 يمكنك القيام بهذا وذلك بإجراء المثال التالي

مثال

```
SQL> SELECT *
      FROM      emp
     WHERE sal BETWEEN 2000 AND 5000;
                                         ↑           ↑
                                         قيمة أدنى   قيمة أعلى
```

في هذا المثال تم استخدام معامل Between مع جملة Where لاستعادة السجلات التي تقع نطاق معين من القيم

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7566	JONES	MANAGER	7839	02/04/81	2975		20
7698	BLAKE	MANAGER	7839	01/05/81	2850		30
7782	CLARK	MANAGER	7839	09/06/81	2450		10
7788	SCOTT	ANALYST	7566	19/04/87	3000		20
7839	KING	PRESIDENT		17/11/81	5000		10
7902	FORD	ANALYST	7566	03/12/81	3000		20

تم اختيار 6 صف

SQL> |

2-2-6 استخدام معامل IN

يمكنك استخدام معامل IN القيام بعمليات البحث عن بنود داخل إحدى القوائم، حيث يمكنك استرجاع السجلات التي قيم عمود فيها (محدد في جملة Where) تساوي إحدى القيم المحددة ضمن مجموعة، فعلى سبيل المثال لإظهار كافة السجلات الخاصة بالموظفين ذوي الأرقام 7369 أو 7499 أو 7521 يتم إجراء المثال التالي

```
SQL> SELECT *
  2 FROM emp
  3 WHERE empno IN (7369, 7499, 7521);
```

مثال 1

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7521	WARD	SALESMAN	7698	22/02/81	1250	500	30
7499	ALLEN	SALESMAN	7698	20/02/81	1600	300	30
7369	SMITH	CLERK	7902	17/12/80	800		20

SQL>

في هذا المثال السابق تم استرجاع السجلات التي فيها العمود empno يساوي إما 7369 أو 7499 أو 7521، فنجد أن IN تكافئ سلسلة مقارنات مربوطة بالمعامل OR، يمكنك إجراء المثال السابق باستخدام OR على النحو التالي

```
SQL> SELECT *
  2 FROM emp
  3 WHERE empno=7369 or empno=7499
or empno=7521;
```

مثال 2

```
SQL> SELECT *
  2 FROM emp
  3 WHERE ename IN ('tarek', 'ayman', 'said');
```

في هذا المثال سيتم استعادة كافة السجلات الخاصة بالموظفين الموجودين في القائمة التي تلي معامل IN، حيث يتم استعادة جميع السجلات من جدول emp عندما يكون ename يساوي 'tarek' Or 'ayman' Or 'said' إما

تلميح

وفيما يلي عدة اعتبارات لابد من مراعاتها عند إعادة استخدام معامل IN مع جملة Where

- ✓ وضع قيم المجموعة بين قوسين .
- ✓ فصل قيم المجموعة عن بعضها بالفاصلة.
- ✓ يمكن وضع القيم بأي ترتيب .
- ✓ عندما تكون قيم المجموعة حرفية أو تاريخ لابد من وضعها بين علامتي تصيص (') .
- ✓ يمكن استخدام IN لعكس الاختيار NOT

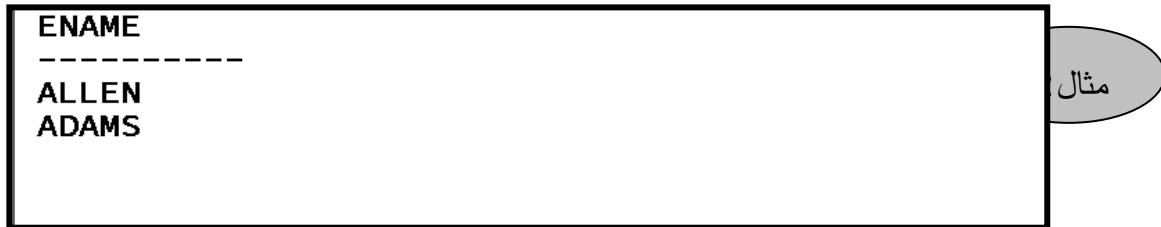
3-2-6 استخدام معامل Like

يمكنك استخدام Like مع جملة Where لاستعادة واسترجاع السجلات التي قيم عمود فيها تشابه سلسلة أحرف نبحث عنها، ليس هذا فحسب بل يمكنك من استرجاع السجلات التي تكون قيم عمود فيها مطابقة جزئياً لمجموعة أحرف نبحث عنها

مثال 1

لاسترجاع جميع الأسماء التي تبدأ بحرف (A) يتم كتابة الكود التالي

```
SQL> SELECT    ename  
  2  FROM      emp  
  3 WHERE     ename LIKE 'A%';
```



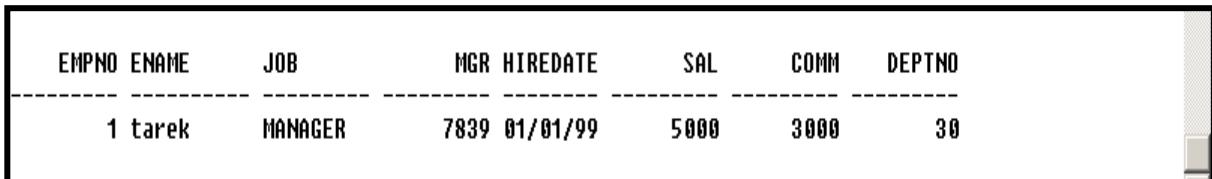
ENAME

ALLEN
ADAMS

مثال

للبحث عن الأسماء التي تشتمل على أحرف are يتم ذلك من خلال كتابة الكود التالي:

```
SQL> SELECT *  
2 FROM emp  
3 WHERE ename LIKE '%are%';
```



EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	tarek	MANAGER	7839	01/01/99	5000	3000	30

مثال 3

لاسترجاع بيانات الاسم والوظيفة والمربى للموظفين الذي تم تعينهم خلال عام 1982 من

شهر يناير إلى شهر ديسمبر يتم كتابة الكود التالي

```
SQL> SELECT ename, job, sal, hiredate  
2 FROM emp  
3 WHERE HIREDATE LIKE '%1982';
```

مثال 4

ENAME	JOB	SAL	HIREDATE
MILLER	CLERK	1300	23/01/82

لاسترجاع جميع الأسماء التي يكون حرف (A) فيها يأخذ الترتيب الثاني يتم كتابة الكود

: التالي:

```
SQL> SELECT    ename
  2  FROM      emp
  3 WHERE     ename LIKE '_A%';
```

ENAME
WARD
MARTIN
JAMES



✓ تستخدم LIKE مع الأعمدة التي يكون نوع بياناتها حرفي .

تمرين

✓ اختصار لأي تتابع من سلسلة أحرف غير معينة مكونة من أي قيمة

وبأي طول

✓ _ اختصار لأي حرف وحيد

4-2-6 استخدام معامل IS NULL

قبل استخدام معامل `IS NULL` لابد من الإجابة على ما هي القيمة `NULL` في جملة `Where`

؟ تستخدم القيمة `NULL` للإشارة إلى عمود لا يحتوي على بيانات، حيث لا تعني القيمة `NULL`

صفر أو مسافة `Blank Space`

مثال 1

لاسترجاع جميع بيانات الموظفين الذين لا يتلقاون حواجز `Commission` يتم

كتابة الكود التالي

```
SQL> SELECT *
      2  FROM      emp
      3 WHERE      comm IS NULL;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17/12/80	800		20
7566	JONES	MANAGER	7839	02/04/81	2975		20
7698	BLAKE	MANAGER	7839	01/05/81	2850		30
7782	CLARK	MANAGER	7839	09/06/81	2450		10
7788	SCOTT	ANALYST	7566	19/04/87	3000		20
7839	KING	PRESIDENT		17/11/81	5000		10
7876	ADAMS	CLERK	7788	23/05/87	1100		20
7908	JAMES	CLERK	7698	03/12/81	950		30
7902	FORD	ANALYST	7566	03/12/81	3000		20
7934	MILLER	CLERK	7782	23/01/82	1300		10

تم اختيار 10 صف

6-3 استخدام عوامل المقارنة الأخرى(Logical Operators)

تتضمن SQL ثلاثة معاملات منطقية بالإضافة للمعاملات التي الإشارة إليها في سابقًا

يمكن استخدامها مع جملة `Where`، كما هو مبين بالجدول التالي

المعامل Operator	المعنى Meaning
AND	و
OR	أو
NOT	النفي

1-3-6 استخدام معامل AND

يستخدم AND مع Where لاستعادة السجلات التي تحقق كافة الشروط المحددة في جملة Where، حيث يعني AND أنه لاستعادة السجلات لابد من تحقق كافة الشروط.

مثال 1

لاسترجاع جميع بيانات الموظفين بالإدارة رقم 10 الذين يتتقاضون مرتبًا أكبر من 3000

ريال يتم كتابة الكود التالي:

```
SQL> SELECT *
  2  FROM    emp
  3 WHERE   sal >3000
  4 AND     deptno=10;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT		17/11/81	5000		10

لاسترجاع جميع بيانات محلية النظم الذين يتتقاضون مرتبًا أكبر من 3000 ريال يتم كتابة

الكود التالي

```
SQL> SELECT ename, job, sal
  2  FROM    emp
  3 WHERE   sal>1000
  4 AND     job='analyst';
```

ENAME	JOB	SAL
SCOTT	ANALYST	3000
FORD	ANALYST	3000

2-3-6 استخدام معامل OR

نستخدم معامل Or مع Where لاستعادة السجلات التي تحقق شرطاً واحداً من عدة شروط، حيث يعني المعامل Or أن استعادة البيانات تتم إذا تحقق أي شرط من الشروط.


 مثال 1

لاسترجاع بيانات الموظفين (الاسم، الوظيفة، المرتب) يتتقاضون مرتبًا أكبر من

3000 ريال أو الوظيفة analyst، يتم كتابة الكود التالي

```
SQL> SELECT ename, job, sal
2 FROM emp
3 WHERE sal>3000
4 OR job='ANALYST';
```

ENAME	JOB	SAL
SCOTT	ANALYST	3000
KING	PRESIDENT	5000
FORD	ANALYST	3000

✓ تم عرض بيانات الموظفين الذين يتتقاضون مرتبًا أكبر من 3000 أو الوظيفة

Analyst

3-3-6 استخدام معامل NOT

استخدام المعامل NOT يمكننا من استثناء سجلات معينة من اختيارنا، حيث يمكننا من استرجاع السجلات المعاكسة لاختيارنا، فعلى سبيل المثال لاختيار كافة السجلات من جدول emp باستثناء التي يكون ال Commission بها قيمة NULL يتم كتابة الكود التالي


 مثال 1

```
SQL> SELECT ename, job
  2 FROM emp
  3 WHERE comm IS NOT NULL;
```

ENAME	JOB
ALLEN	SALESMAN
WARD	SALESMAN
MARTIN	SALESMAN
TURNER	SALESMAN


 مثال 2

```
SQL> SELECT ename, job
  2 FROM emp
  3 WHERE job NOT IN
ana ('ANALYST', 'ACCOUNTANT', 'ENGINEER');
    engineer, accountant يتم كتابة الكود التالي :
```

ENAME	JOB
SMITH	CLERK
ALLEN	SALESMAN
WARD	SALESMAN
JONES	MANAGER
MARTIN	SALESMAN
BLAKE	MANAGER
CLARK	MANAGER
KING	PRESIDENT
TURNER	SALESMAN
ADAMS	CLERK
JAMES	CLERK
MILLER	CLERK

12 ROW Selected

تلميح

✓ يتم استخدام معامل NOT مع المعاملات الأخرى مثل

(BETWEEN, LIKE, and NULL)

✓ وإليك التلميح التالي:

```
.... Where sal NOT BETWEEN 3000 AND 4000
.... Where ename NOT LIKE '%T%'
.... Where job NOT IN ('analyst', 'accountant')
.... Where comm IS NOT NULL
```

6-4 قواعد أسبقية المعاملات

الجدول التالي يوضح قواعد أسبقية المعاملات

Order	Evaluated الترتيب	Operator المعامل
1		جميع عوامل المقارنة
2		NOT
3		AND
4		OR

تلميح

✓ كما تعلمنا سابقاً يمكننا تغيير تسلسل أسبقية المعاملات باستخدام الأقواس (Parentheses)، حيث تأخذ الأقواس الأسبقية في التعبيرات الحسابية

مثال 1

```
SQL>SELECT ename, job, sal
  2 FROM emp
  3 WHERE job='ANALYST' ← الثاني
  4 OR    job='MANAGER' ← الأول
  5 AND    sal >3000;
```

الثاني

الأول

Job تكون MANAGER و Sal أكبر من 3000 ريال

✓ الشرط الأول هو

ANALYST Job تكون

✓ الشرط الثاني هو

تلميح

ENAME	JOB	SAL
SCOTT	ANALYST	3000
FORD	ANALYST	3000

7 - ترتيب النتائج باستخدام Order by

ترتيب السجلات باستخدام Order by وفقاً لقيم العمود الذي نحدده ترتيباً تصاعدياً أو تناظرياً، ويمكن الترتيب بناءً على قيمة عمود أو عدة أعمدة، وعند كتابة جملة Order by يجب إتباع الآتي:

✓ تابع جملة Order by مباشرة في حالة عدم from {table name}

✓ وجود Where {Condition(s)}

✓ تابع جملة Order by مباشرة . Where {Condition(s)}

✓ يوجد لدينا خيارات عند استخدام Order by

1. يسمح هذا الخيار Order by {column name} ASC :

بالترتيب التصاعدي

2. يسمح هذا الخيار Order by {column name} desc :

بالترتيب التنازلي.

✓ يمكن الترتيب على أساس عدة أعمدة، ويجب الفصل بين أسماء الأعمدة بفواصل

(,)

✓ يوجد لدينا طريقتان لكتابة جملة Order by وهما

1. عن طريق اسم العمود

2. عن طريق رقم العمود النسبي (موقعه في جملة SELECT)

1-7 الترتيب التصاعدي باستخدام Order by

مثال 1

لاستعادة بيانات جميع الموظفين مرتبة علي أساس تاريخ التعيين، يتم كتابة الكود التالي

```
SQL> SELECT *
  2 FROM      emp
  3 ORDER BY hiredate asc;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17/12/80	800		20
7499	ALLEN	SALESMAN	7698	20/02/81	1600	300	30
7521	WARD	SALESMAN	7698	22/02/81	1250	500	30
7566	JONES	MANAGER	7839	02/04/81	2975		20
7698	BLAKE	MANAGER	7839	01/05/81	2850		30
7782	CLARK	MANAGER	7839	09/06/81	2450		10
7844	TURNER	SALESMAN	7698	08/09/81	1500	0	30
7654	MARTIN	SALESMAN	7698	28/09/81	1250	1400	30
7839	KING	PRESIDENT		17/11/81	5000		10
7900	JAMES	CLERK	7698	03/12/81	950		30
7902	FORD	ANALYST	7566	03/12/81	3000		20
7934	MILLER	CLERK	7782	23/01/82	1300		10
7788	SCOTT	ANALYST	7566	19/04/87	3000		20
7876	ADAMS	CLERK	7788	23/05/87	1100		20

تم اختيار 14 صف

SQL> |

مثال 2

```
SQL>SELECT *
  2 FROM      emp
  3 ORDER BY hiredate DESC;
```

1-7 الترتيب التنازلي باستخدام Order by

لاستعادة بيانات جميع الموظفين مرتبة تنازلياً علي أساس تاريخ التعيين، يتم كتابة الكود

التالي

```
SQL> SELECT *
  2 FROM      emp
  3 ORDER BY hiredate desc;
```



تلميح

✓ الترتيب الافتراضي في جملة Order by هو الترتيب التصاعدي ملماً نصرح

عكس ذلك بالكلمة DESC

✓ يتم عرض القيم null مؤخراً عند الترتيب التصاعدي، والعكس يتم ظهر القيم

أولاً عند الترتيب التنازلي.

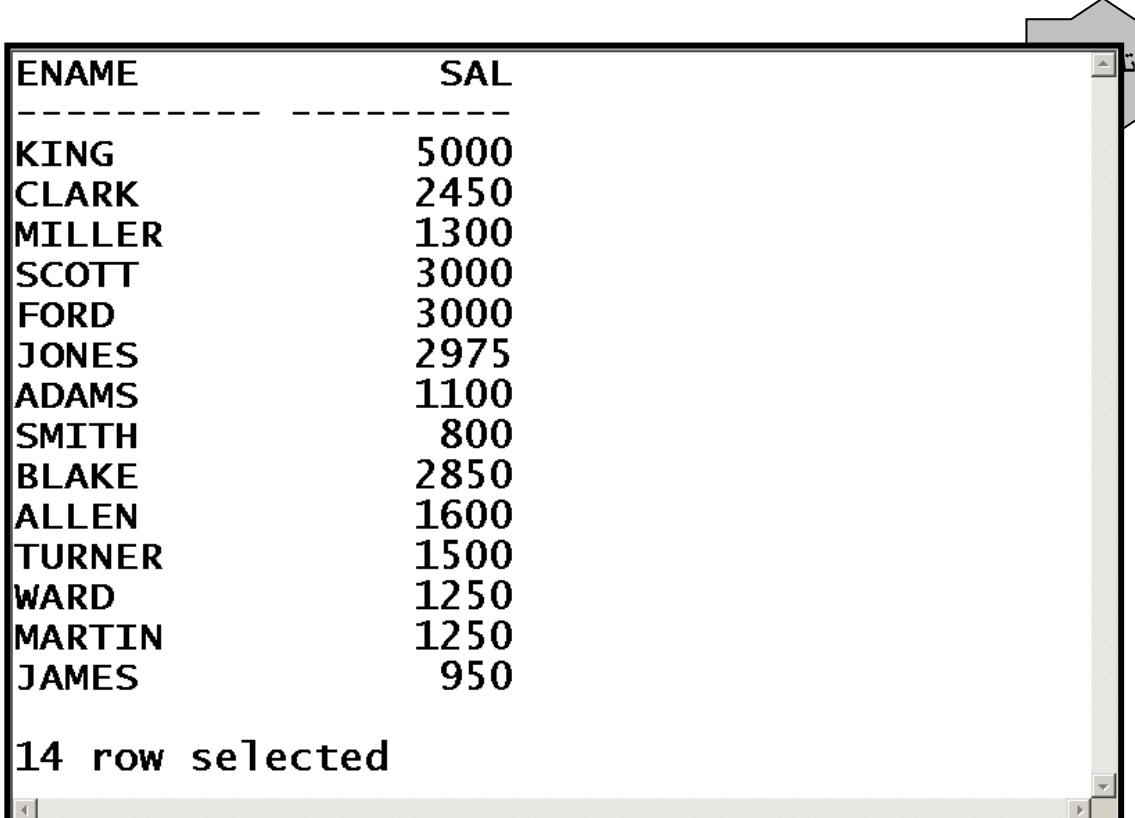


مثال 3

يمكنك إجراء الترتيب باستخدام Order by على أساس قيم أعمدة ليست بجملة

كما بهذا المثال SELECT

```
SQL> SELECT      ename, sal
  2  FROM        emp
  3 ORDER BY    deptno, sal DESC;
```



ENAME	SAL
KING	5000
CLARK	2450
MILLER	1300
SCOTT	3000
FORD	3000
JONES	2975
ADAMS	1100
SMITH	800
BLAKE	2850
ALLEN	1600
TURNER	1500
WARD	1250
MARTIN	1250
JAMES	950

14 row selected

- ✓ يمكننا إجراء الترتيب باستخدام Order by على أساس قيم أكثر من عمود.

3- الترتيب باستخدام A Column Alias

يمكننا إجراء الترتيب في جملة Order by باستخدام A column Alias كما

بالمثال التالي:

مثال

```
SQL> SELECT      empno, ename, sal*12 total
  2 FROM        emp
  3 ORDER BY  total;
```

EMPNO	ENAME	TOTAL
7369	SMITH	9600
7900	JAMES	11400
7876	ADAMS	13200
7521	WARD	15000
7654	MARTIN	15000
7934	MILLER	15600
7844	TURNER	18000
7499	ALLEN	19200
7782	CLARK	29400
7698	BLAKE	34200
7566	JONES	35700
7788	SCOTT	36000
7902	FORD	36000
7839	KING	60000

14 row selected

8- القيمة Null وتأثيراتها

تستخدم القيمة null للإشارة إلى عمود لا يحتوي على بيانات بمعنى وجود قيم فارغة داخل الحقول ومن أشهر عيوب القيمة null صعوبة إجراء أي عمليات حسابية على سجلات تحتوى بعضها على قيم فارغة

1-8 استخدام القيمة Null

1) تستخدم القيمة null للإشارة إلى عمود لا يحتوي على بيانات، ولا تعني القيمة

null الصفر أو المسافة الفارغة blank space

2) عند استخدام القيمة null في التعبيرات الحسابية يكون الناتج null

مثال 1

```
SQL> SELECT ename, sal, comm  
2 FROM emp;
```

في هذا المثال نجد أن العمود comm يحتوي قيمًا من النوع null

ENAME	SAL	COMM
SMITH	800	
ALLEN	1600	300
WARD	1250	500
JONES	2975	
MARTIN	1250	1400
BLAKE	2850	
CLARK	2450	
SCOTT	3000	
KING	5000	
TURNER	1500	0
ADAMS	1100	
JAMES	950	
FORD	3000	
MILLER	1300	

14 row selected

مثال 2

```
SQL> select ename,sal,comm,sal+comm
  2 from emp
  3 WHERE ename='SMITH';
```

ENAME	SAL	COMM	SAL+COMM
SMITH	800		

في هذا المثال تم تناول العمليات الحسابية على القيم null، حيث أن العمود comm الخاص بالموظف SMITH يأخذ القيمة null، وكما تعلمنا أن ناتج أي عملية حسابية طرفاها القيمة null يكون null

9 - استخدام محرر النصوص في البرنامج SQL*PLUS

بعد إدخال تعليمات ما في محوّل SQL فإنها تبقى في الذاكرة المؤقتة (BUFFER)، تقوم بكتابه التعليمة `List` أو الحرف `L` لإظهار آخر تعليمات أرسلت إلى ORACLE، ويمكن إعادة تنفيذها بإدخال الكلمة `RUN` أو الحرف `r`، ويمكن تعديل التعليمات التي تم إدخالها إلى الذاكرة المؤقتة ، حيث يمكنك `sql*plus` سلسلة من التعليمات التي تسمح للمستخدم بتعديل محتوى التعليمات المخزنة في الذاكرة المؤقتة كما هو موضح في الجدول التالي :

مثال	الوظيفة	التعليمية
<code>l</code>	إظهار محتوى الذاكرة المؤقتة	<code>l</code>
<code>I</code>	السماح للمستخدم بالبدء بإدخال نص بعد السطر الحالي	<code>I</code>
<code>/ RUN</code>	تنفذ التعليمية المخزنة حالياً في الذاكرة المؤقتة	<code>/run</code> أو <code>R</code>
<code>Save c:\t.sql</code>	حفظ التعليمية الحالية في ملف	<code>Save filename</code>
<code>get c:\t.sql</code>	تحميل محتوى ملف ما إلى الذاكرة المؤقتة	<code>Get filename</code>
<code>start c:\t.sql</code>	تحميل الملف إلى الذاكرة المؤقتة وتنفيذ التعليمات الموجودة به	<code>Start filename</code> أو <code>@filename</code>
<code>edit c:\t.sql</code> أو <code>ed</code>	تشغيل محرر النصوص الخارجي مثل المفكرة، فإذا تم تحديد إسم الملف يتم فتح محرر النصوص متضمناً الملف	<code>Edit filename</code>

الأسئلة

المجموعة الأولى:

ضع علامة (✓) أمام العبارة الصحيحة وعلامة (✗) أمام العبارة الخاطئة مع تصحيح الخطأ.

(1) لغة DDL تستطيع من خلالها معالجة البيانات الموجودة في قاعدة البيانات من إدراج

البيانات وتحديثها أو تحديد أو حذف البيانات.

(2) تمثل عبارة Select * طريقة مختصرة لاسترجاع أعمدة محددة من الجدول.

3) تظهر الأعمدة المطلوبة بدءاً من اليسار إلى اليمين بنفس ترتيب كتابتها في عبارة

.Select

4) تتجاهل SQL*Plus المسافات Blank Spaces قبل وبعد المعاملات

الحسابية.

4) إذا احتوي alias على مسافات أو أحرف وعلامات خاصة لابد من وضعه بين

علامتي تصيص (" .")

5) تأتي جملة Where قبل جملة From

6) يلي Where شرط أودعة شروط.

7) نستخدم معامل Between مع جملة Where لاستعادة السجلات التي تقع نطاق معين

من القيم.

8) يمكنك معامل LIKE القيام بعمليات البحث عن بنود داخل إحدى القوائم، بينما يمكنك

استخدام IN استعادة واسترجاع السجلات التي قيم عمود فيها تشبه سلسلة أحرف نبحث

عنها

9) تستخدم معامل LIKE مع الأعمدة التي يكون نوع بياناتها رقمي.

10) يعني معامل AND أنه لاستعادة السجلات لابد من تحقق كافة الشروط

المجموعة الثانية:

أسئلة لتدريب الطلاب على أسئلة الاختيار المتعدد:

(1) المصطلح DDL هو ..

1. اختصار .data definition language

2. بناء SQL المستخدم لتعريف البيانات في قاعدة البيانات.

3. بناء SQL المستخدم لمعالجة البيانات في قاعدة البيانات.

. 1&2 . 4

(2) المصطلح DML هو :-

1. بناء SQL المستخدم لمعالجة البيانات في قاعدة البيانات .

. اختصار ل Data Manipulation Language

3. بناء SQL المستخدم لتعريف البيانات .

. 2&1 . 4

(3) الأمر update :-

1. من أوامر DML .

2. يستخدم لتحديث البيانات في جدول .

3. من أوامر DQL .

. 1&2 . 4

(4) الأمر Insert :-

1) من أوامر DML .

2) من أوامر DDL .

3) يستخدم في إدراج البيانات في جدول .

. 1&3 (4)

الأمر 5: drop

1. من أوامر DML.
2. من أوامر DDL.
3. يستخدم لحذف حقل من جدول.
4. 2 & 3

الأمر 7: Revoke من:

- 1- أوامر DDL.
- 2- أوامر DML.
- 3- أوامر DQL.
- 4- أوامر PL/SQL

الأمر 8: GRANT من:

1. أوامر DDL.
2. أوامر DML.
3. أوامر DQL.
4. أوامر PL/SQL

الأمر 9: insert من :

1. أوامر DML.
2. أوامر DQL.
3. أوامر DDL.

.PL/SQL أوامر 4.

الأمر **Truncate** يستخدم في:

- 1) حذف الأعمدة في الجدول.
- 2) حذف كافة الصفوف الموجودة في الجدول.
- 3) حذف عمود واحد.
- 4) حذف صف واحد.

11) بفرض أن لديك الجدول emp التالي :

empno	name	Sal
1	Ali	1000
2	Hani	2000
3	Mohammed	1500
4	Ayman	1200

يكون ناتج الأوامر التالية هي :

select * from emp where sal > 1500 ;

1) سجل 1

2) سجل 2

(3) كل السجلات

4 سجل (4)

```
select * from state where empno not in (1 , 2) ;
```

4 سجل 3 و (1)

4 سجل 1 و (2)

3 سجل (3)

4 سجل (4)

```
select * form state where sal between 1000 and 1300 ;
```

4 سجل 2 و (1)

4 سجل 1 و (2)

3 سجل (3)

4 سجل (4)

```
select * from state where name like ' A% ' ;
```

4 سجل 2 و (1)

4 سجل 1 و (2)

3 سجل 2 و (3)

4 سجل (4)

```
select*from state where sal not between 1000 and 1500 ;
```

2) سجل

(2) سجل 1 و 4

3) سجل

(4) سجل 4

الفصل الثاني

تقنيات لغة SQL

الأهداف:

إعطاء الطالب صورة موضحة عن أساسيات الربط بين الجداول في SQL، استخدام الدوال having، Functions مع جملة SELECT، بالإضافة إلى إمكانية إنشاء استعلام فرعي sub query group by

المحتويات:

1. الرابط بين جدولين أو أكثر

multi-row functions 2. استخدام

Single Row Functions 3. استخدام

Group by clause 4. استخدام

Having clause 5. استخدام

sub query 6. استخدام وعمل استعلام فرعي

ماذا سنتعلم في هذا الفصل:

في نهاية هذا الفصل يكون الطالب قد اكتسب المهارات والمعرفات التالية:

1. الربط بين جدولين أو أكثر
2. التعرف على أنواع العلاقات بين الجداول
3. استعادة بيانات من جدولين أو أكثر
4. استخدام group functions
5. استخدام Single Row Functions
6. إنشاء مجموعات من السجلات باستخدام Having, group by
7. استخدام وعمل استعلام فرعي Sub query

1- الرابط بين جدولين أو أكثر

كثيراً ما نحتاج إلى استرجاع بيانات من جدولين أو أكثر في استعلام واحد Query، لإجراء مثل ذلك يتطلب منا إنشاء علاقة بين الجداول، حيث تكمن القوة الحقيقية لقاعدة بيانات علائقية في قدرتها على استرجاع سجلات من جداول مختلفة في استعلام واحد، وتوجد لدينا العديد من الأسباب التي تدعونا إلى إنشاء علاقة بين الجداول منها:

- 1) دمج الأعمدة من جدولين أو أكثر.
- 2) اختيار أعمدة موجودة في جدول واحد بناء على شرط يطبق على عمود آخر (self join).

1-1 استرجاع بيانات من جدولين أو أكثر (Multiple Tables)

عند استرجاع بيانات من جدولين أو أكثر هناك عدة اعتبارات يجب مراعاتها

- 1) وضع أسماء الأعمدة في جملة Select.
- 2) وضع أسماء الجداول في جملة From، على أن يتم الفصل بينها بالفاصة.
- 3) إضافة شرط الرابط في جملة Where.
- 4) الصيغة العامة لاسترجاع بيانات من جدولين

```
SELECT      table1.column, table2.column
FROM        table1, table2
WHERE       table1.column1 = table2.column2;
```

مثال 1

```
SQL>SELECT emp.empno, emp.ename, dept.dname
  2 FROM      emp, dept
  3 WHERE     emp.deptno=dept.deptno;
```

في هذا المثال تم استرجاع بيانات من جدولين dept, emp بينهما علاقة، حيث تم كتابة إسم الجدول قبل إسم العمود مع الفصل بينهما بنقطة في جملة select، كتابة الجداول مصدر البيانات في جملة from، مع كتابة شرط الربط في جملة where.

EMPNO	ENAME	DNAME
7369	SMITH	RESEARCH
7499	ALLEN	SALES
7521	WARD	SALES
7566	JONES	RESEARCH
7654	MARTIN	SALES
7698	BLAKE	SALES
7782	CLARK	ACCOUNTING
7788	SCOTT	RESEARCH
7839	KING	ACCOUNTING
7844	TURNER	SALES
7876	ADAMS	RESEARCH
7900	JAMES	SALES
7902	FORD	RESEARCH
7934	MILLER	ACCOUNTING

14 row selected

تلميح

في المثال السابق نجد أن أعمدة الرابط لها نفس الاسم deptno، وهذا يسبب مشاكل مع أوراكل حيث لا يعرف لأي جدول تتبع هذه الأعمدة ولذلك تم وضع اسم الجدول قبل اسم العمود، مع الفصل بينهما بنقطة.

٢-١ أنواع العلاقات

١. الرابطة المتكافئة **equijoin**: لابد أن كل سجل في الجدول الرئيس الأب

يقابله سجل أو سجلات في الجدول المرتبط به الأبن

2. الرابطة غير المتكافئة **nonequi join**: لا تمتلك الجداول أعمدة وقيماً مشتركة

3. الرابطة الخارجية **outer join**: تمتلك الجداول المرتبطة أعمدة مشتركة، ولكن السجلات المرتبطة يمكن أن تحتوي قيم مشتركة ويمكن أن لا تحتوي، بمعنى أن الجدول الرئيس الأب يحتوي سجل أو سجلات لا يقابلها سجل أو سجلات في الجدول المرتبط به الابن

4. الروابط الذاتية **Self join**: يمكننا عمل رابطة ذاتية عند ربط الجدول مع نفسه

5. الرابطة الديكارتية **Cartesian joins**:

للحصول على الرابطة الديكارتية قم بحذف شرط الربط بين الجدولين في جملة Where

سيقوم أوراكل بربط كل سجل في الجدول الأول مع كل سجل في الجدول الثاني مع نفسه

1-2-1 الرابطة المتكافئة

فعلي سبيل المثال إذا أردت تحديد اسم الإدارة لكل موظف فما عليك سوى المقارنة بين قيم فإذا عمود deptno في جدول الموظفين وقيم deptno في جدول dept كانت متساوية بمعنى أن جدول dept لا يوجد به إدارة ليس بها عاملين، بمعنى أن كل الإدارات يتم اختيارها في استعلام يربط الجدولين، إن مثل تلك العلاقة تسمى بالرابطة المتكافئة.

مثال 1

```
SQL>SELECT emp.empno, emp.ename, emp.deptno,
  Dept.deptno, dept.loc
  FROM emp, dept
 WHERE emp.deptno=dept.deptno;
```

EMPNO	ENAME	DEPTNO	DEPTNO	LOC
7369	SMITH	20	20	DALLAS
7499	ALLEN	30	30	CHICAGO
7521	WARD	30	30	CHICAGO
7566	JONES	20	20	DALLAS
7654	MARTIN	30	30	CHICAGO
7698	BLAKE	30	30	CHICAGO
7782	CLARK	10	10	NEW YORK
7788	SCOTT	20	20	DALLAS
7839	KING	10	10	NEW YORK
7844	TURNER	30	30	CHICAGO
7876	ADAMS	20	20	DALLAS
7900	JAMES	30	30	CHICAGO
7902	FORD	20	20	DALLAS
7934	MILLER	10	10	NEW YORK
14 row selected				

- في هذا المثال نجد أن جدول dept لا يوجد به إدارة ليس بها عاملين في جدول emp، حيث في هذا الاستعلام تم استرجاع بيانات لجميع الإدارات.
- حيث أن عمود deptno له نفس الاسم في كلا الجدولين، تم وضع اسم الجدول قبل اسم العمود، مع الفصل بينهما بنقطة.
- يطلق على العلاقة simple join أو equijoin أيضاً inner join على العلاقة.

1-2-1 استخدام أكثر من شرط في جملة WHERE

يمكننا عند استرجاع بيانات من أكثر من جدول، يمكننا إضافة أكثر من شرط في جملة الشرط بعد شرط الرابط، كما بالمثال التالي، فمثلاً لاسترجاع بيانات محلية النظم من الرقم، الاسم، الوظيفة، رقم الإدارة، موقع الإدارة، لابد من إضافة شرط في `where` كما بالمثال التالي:

```
SQL>SELECT empno, ename, emp.job, emp.deptno, loc
      FROM emp, dept
     WHERE emp.deptno=dept.deptno
       And job='ANALYST';
```

EMPNO	ENAME	JOB	DEPTNO	LOC
7788	SCOTT	ANALYST	20	DALLAS
7902	FORD	ANALYST	20	DALLAS

Where في جملة TABLE Aliases

كما سبق وتعلمنا أنه يمكن استخدام اسم اعتبري للعمود، يمكننا أيضاً اسم اعتبري للجدول فعلى سبيل المثال يمكننا إجراء مثل 1 السابق `Table Aliases` باستخدام الأسماء الاعتبارية للجداول

```
SQL>SELECT e.empno, e.ename, e.deptno,
      d.deptno, d.loc
      FROM emp e, dept d
     WHERE e.deptno=d.deptno;
```



للمزيد

هناك عدة اعتبارات يجب مراعاتها عند استخدام الأسماء الاعتبارية

(1) الاسم الاعتباري للجدول لا يزيد عن 30 حرفاً.

(2) لابد أن يستخدم الاسم الاعتباري أولاً في جملة `WHERE`

(3) يفضل أن يكون الاسم الاعتباري معبراً

EMPNO	ENAME	DEPTNO	DEPTNO	LOC
7369	SMITH	20	20	DALLAS
7499	ALLEN	30	30	CHICAGO
7521	WARD	30	30	CHICAGO
7566	JONES	20	20	DALLAS
7654	MARTIN	30	30	CHICAGO
7698	BLAKE	30	30	CHICAGO
7782	CLARK	10	10	NEW YORK
7788	SCOTT	20	20	DALLAS
7839	KING	10	10	NEW YORK
7844	TURNER	30	30	CHICAGO
7876	ADAMS	20	20	DALLAS
7900	JAMES	30	30	CHICAGO
7902	FORD	20	20	DALLAS
7934	MILLER	10	10	NEW YORK

14 row selected

3-1-2-1 استرجاع بيانات من عدة جداول (More than two tables)

أحياناً ما نريد استرجاع بيانات من أكثر من جدولين، فعلى سبيل المثال نريد استرجاع

بيانات الاسم ورقم الطلبيه والأصناف وإجمالي الأصناف، إجمالي الطلبيات للعميل

Tarek customer, ord, item يمكن استرجاع تلك البيانات من ثلاثة جداول هي

item، كما بالمثال التالي

```
SQL>SELECT c.name, o.ord, i.item, i.itemtot, o.total
      FROM customer c,ord o,item i
     WHERE c.custid=o.custid
       And o.ordid=i.ordid
       And c.name='Tarek';
```

1-2-2 الرابطة غير المكافئة nonequi join

نادراً ما نحتاج إلى ربط سجلات من جدولين لا يمتلكان أعمدة ولاقيما مشتركة. إن مثل تلك الرابطة تسمى رابطة غير مكافئة، كما بالمثال التالي

emp

EMPNO	ENAME	SAL
1	tarek	4000
2	ayman	2850
3	said	2450
4	ali	2998
5	ahmed	1350
6	helal	1700
7	ebrahim	1600
...		
25 rows selected.		

salgrade

RADAE	LOSAL	HISAL
1	800	1300
2	1301	1400
3	1401	2500
4	2500	9999

المرتب في جدول emp يقع بين
 أقل مرتب وأكبر مرتب في جدول
 SALGRADE



تلميح

✓ العلاقة بين الجدولين في عمود `sal` في جدول `emp` ينحصر بين عمودي

.`Salgrade` في جدول `Losal,hisal`

✓ في الروابط غير المكافئة لا تستخدم معامل المساواة



مثال

```
SQL>SELECT emp.ename, emp.sal, salgrade.graue
      FROM emp, salgrade
     WHERE emp.sal
       BETWEEN salgrade.losal AND salgrade.hisal;
```

File Edit Format Help

SMITH	800	1
ADAMS	1100	1
JAMES	950	1
WARD	1250	2
MARTIN	1250	2
MILLER	1300	2
ALLEN	1600	3
TURNER	1500	3
JONES	2975	4
BLAKE	2850	4
CLARK	2450	4
SCOTT	3000	4
FORD	3000	4
KING	5000	5

14 row selected

3-2-3 الرابطة الخارجية outerjoin

تمتلك الجداول المرتبطة أعمدة مشتركة، ولكن السجلات في الجداول المرتبطة يمكن أن تحتوي فيهم مشتركة ويمكن أن لا تحتوي، فعلى سبيل المثال إذا احتوى جدول dept على إدارة لا يوجد بها عاملين، يعني ذلك أن تلك الإدارة لن يتم اختيارها في استعلام يربط بين الجدولين، فعلى سبيل المثال في الشكل التوضيحي التالي نجد أن لا يوجد موظفين في .

الإدارة ENGINEERING

emp		dept	
ENAME	DEPTNO	DEPTNO	DNAME
Tarek	1	1	ACCOUNTING
Ayman	1	2	SALES
Said	3	3	RESEARCH
Ali	2	...	
...		10	ENGINEERING

↑ لا يوجد موظفين في الإدارة
ENGINEERING

فإذا قمنا بكتابه الكود التالي:

```
SQL> SELECT e.ename, d.deptno, d.dname
  2 FROM emp e, dept d
  3 WHERE e.deptno = d.deptno
```

ENAME	DEPTNO	DNAME
SMITH	20	RESEARCH
ALLEN	30	SALES
WARD	30	SALES
JONES	20	RESEARCH
MARTIN	30	SALES
BLAKE	30	SALES
CLARK	10	ACCOUNTING
SCOTT	20	RESEARCH
KING	10	ACCOUNTING
TURNER	30	SALES
ADAMS	20	RESEARCH
JAMES	30	SALES
FORD	20	RESEARCH
MILLER	10	ACCOUNTING

14 row selected

نجد أن هناك سجلات لا تحقق شرط الربط (WHERE e.Deptno = d.deptno) وبالتالي لن يتم استرجاعها في نتيجة الاستعلام، وحيث لا يوجد موظفين بالإدارة ENGINEERING، وبالتالي لن تظهر في نتيجة الاستعلام طبقاً للEquijoin،لكي نعالج ذلك لابد من إنشاء رابطة خارجية وذلك بوضع إشارة الجمع (+) محاطة بقوسین وذلك بعد إسم الجدول الذي يحتوي السجلات التي لا يقابلها سجل محدد شرط الربط في جملة Where وذلك للجدول الذي يحتوي السجلات التي لا يقابلها سجل أو سجلات في الجدول المرتبط.

مثال

```
SQL>SELECT e.ename, d.deptno, d.dname, d.loc
  FROM emp e, dept d
 WHERE e.deptno(+) = d.deptno;
```

في هذا المثال الإدارة ENGINEERING سيتم عرضها في نتيجة هذا الاستعلام، وذلك

من خلال إنشاء رابطة خارجية وذلك بوضع إشارة الجمع (+) محاطة بقوسین بعد اسم محدد شرط الربط في جملة `Where` وذلك للجدول الذي يحتوي السجلات التي لا يقابلها سجل أو سجلات في الجدول المرتبط وهو في مثالنا جدول `emp`.

Selfjoin الذاتية 4-2-1

يمكنا عمل الرابطة الذاتية عند ربط جدول ما مع نفسه، فعندما تربط جدولًا مع نفسه سوف تكتب اسم نفس الجدول عدة مرات في جملة `Where`، فيجب علينا استخدام أسماء اعتبارية `Aliases`، وفي المثال الموضح بالشكل التالي حالة جدول `emp` الذي يحتوي على عمود `(mgr)` حيث يحدد رقم المدير الرقم في عمود المفتاح الرئيسي `empno` بجدول `emp`، وذلك توجد علاقة من `Selfjoin` بين عمود `mgr` وعمود `empno`.

EMP (WORKER)			EMP (MANAGER)	
EMPNO	ENAME	MGR	EMPNO	ENAME
1	tarek		1	tarek
2	ayman	1	1	tarek
3	said	1	1	tarek
4	ali	1	2	ayman
5	ahmed	2	2	ayman
6	ebrahim	2		

"`MANAGER` في جدول `WORKER` نساوي `EMPNO` في جدول `MGR`"

مثال

```
SQL>SELECT worker.ename,manager.ename
  FROM emp worker, emp manager
 WHERE worker.mgr = manager.empno
```

ENAME	ENAME
SMITH	FORD
ALLEN	BLAKE
WARD	BLAKE
JONES	KING
MARTIN	BLAKE
BLAKE	KING
CLARK	KING
SCOTT	JONES
TURNER	BLAKE
ADAMS	SCOTT
JAMES	BLAKE
FORD	JONES
MILLER	CLARK
13 row selected	

1-2-4 الرابطة الديكارتية Cartesian joins

للحصول على الرابطة الديكارتية قم بحذف شرط الربط بين الجدولين في جملة Where، سيقوم أوراكل بربط كل سجل في الجدول الأول مع كل سجل في الجدول الثاني، هذا يجعل أوراكل يقوم بإنشاء ومعالجة كمية كبيرة جداً من البيانات لا قيمة لها، فعلى سبيل المثال إذا كان جدول dept يحتوي 10 سجلات وجدول emp يحتوي 30 سجل، فإذا تم حذف شرط الربط من جملة

في نتيجة الاستعلام تكون (30 * 10) وهي 300 سجل ، ولحساب السجلات المسترجعة

من رابطة ديكارتية للجدولين dept , emp يتم كتابة الاستعلام التالي

```
SQL>SELECT count(*) from dept,emp;
```

COUNT(*)
300

2 - استخدام الدوال

1-2 استخدام Multi row Functions

تزودنا SQL بعده دوال تجميع Group Functions يمكن أن تتفذ على مجموعة من السجلات وهي:

الاستخدام	الدالة
عدد السجلات في جدول ما	COUNT
تحديد القيمة العظمى لعمود ما	MAX
تحديد القيمة الدنيا لعمود ما	MIN
المتوسط الحسابى لعمود ما	AVG
الانحراف المعياري لعمود ما	STDDEV
المجموع الكلى لعمود ما	SUM
حساب التغير لعمود ما	VARIANCE

تقوم الدوال التجميعية Group Functions بمعالجة قيم العمود المختار من الجدول وتقدم

النتيجة في شكل قيمة وحيدة تخص العمود المختار، ويجب عليك إتباع التعليمات التالية عند

استخدام Group function

1) وضع إسم العمود بين قوسين () بعد الدالة مباشرة

```
Select MAX(sal) From emp;
```

2) يمكن لجميع الأعمدة في جملة Select أن تكون كلها دوال تجميعية

```
Select MAX(sal), MIN(SAL), SUM(sal);
```

3) استخدام دالة تجميعية أخرى غير مسموح به في SQL

```
Select MAX(avg(sal)) from emp;
```

4) الدوال التجميعية تتجاهل القيم الفارغة NULLVALUES

5) يمكن استخدام الدوال التجميعية ضمن التعبيرات الحسابية

1-1-2 استخدام count

نستخدم الدالة count لحساب عدد السجلات التي تحتوي قيمةً غير فارغة (not null)

في العمود المحدد في المجموعة، فمثلاً لحساب عدد السجلات في جدول emp الذي

يملك قيمةً غير فارغة في العمود ename يتم كتابة الكود التالي:

```
SQL>SELECT count (ename) from emp;
```

COUNT(ENAME)	للمزيد
14	

✓ تم وضع العمود المراد حساب عدد سجلاته بين قوسين بعد الدالة count.

✓ لا يتم عد السجلات التي تحتوي على قيمة فارغة null.

1-1-1-2 استخدام count(*)

عندما نستبدل إسم العمود بالنجمة (*) فعنده تقوم الدالة count بحساب عدد السجلات بما

فيها السجلات التي تحتوي على قيمة فارغة، فعلى سبيل المثال:

```
SQL> SELECT COUNT (*)  
2 FROM emp  
3 WHERE deptno = 10;
```

COUNT(*)
3

2-1-1-2 استخدام count مع distinct

يمكننا استخدام الكلمة count مع distinct، حيث تقوم باستبعاد السجلات المكررة، فعلى

سبيل المثال لحساب عدد السجلات في جدول emp مع عدم استبعاد القيم الفارغة في العمود

ename يتم كتابة الكود التالي:

```
SQL>SELECT count (distinct (ename)) from emp;
```

2-1-2 استخدام max

نستخدم الدالة MAX لحساب أكبر قيمة في مجموعة ما، يمكن استخدام الدالة MAX مع القيم

العددية أو النصية أو التاريخ والوقت، فعلى سبيل المثال:

```
SQL> SELECT MAX(SAL), MAX(hiredate), MAX(ename)
  2 FROM emp;
```

للمزيد

- ✓ يمكن استخدام `max` مع أي نوع من البيانات.
- ✓ عند استخدام `MAX` مع البيانات الحرفية تعيد أعلى قيمة ASCII، وعند استخدامها مع الأعداد العددية تعيد أعلى قيمة جذرية، وعند استخدامها مع التاريخ والوقت تعيد القيمة الأحدث في العمود.

✓ تهمل الدالة `MAX` القيم `NULL`.

3-1-2 استخدام MIN

تعيد الدالة `MIN` أصغر قيمة لا تساوي `NULL`، كما أنها تستخدم مع أي نوع من البيانات، فعلى سبيل المثال نستخدم الدالة `MAX` لتحديد أقل مرتب في الإدارة رقم 10

```
SQL> SELECT MIN(SAL)
  FROM emp
 WHERE deptno=10;
```

MIN(SAL)

1300

4-1-2 استخدام AVG

نستخدم `AVG` لحساب المتوسط الحسابي لعمود ما، حيث يعيد `AVG` القيمة الوسطى لجميع القيم التي لا تساوي `NULL` في العمود، وهناك عدة اعتبارات يجب مراعاتها عند استخدام `AVG`

- ✓ يستخدم AVG مع القيم العددية فقط.
- ✓ يهمل AVG القيمة NULL عند حسابه للقيمة الوسطي.
- ✓ يهمل AVG القيم المتكررة في العمود عند حسابه للقيمة الوسطي وذلك إذا صرحت بكتابية DISTINCT قبل إسم العمود.

مثال 1

يمكننا حساب المتوسط الحسابي لعمود SAL في جدول EMP، وذلك بكتابة الكود التالي:

```
SQL>SELECT      AVG (SAL)
   FROM emp ;
```

AVG(SAL)

2073.2143

مثال 2

يمكننا حساب كل أعلى قيمة وأدنى قيمة والمتوسط الحسابي لعمود SAL بالكود التالي:

```
SQL> SELECT MAX(SAL) , MIN(SAL) , AVG(SAL)
   FROM emp ;
```

مثال 3

MAX(SAL)	MIN(SAL)	AVG(SAL)
-----	-----	-----
5000	800	2073.2143

ي

مثال 3

يمكننا حساب المتوسط الحسابي لعمود COMM في جدول EMP، وذلك بكتابة الكود التالي

```
SQL>SELECT      AVG (SAL)
      FROM    emp   ;
```

AVG(COMM)

550

نلاحظ في هذا المثال أن avg تتجاهل القيم null.

4-1-2 استخدام SUM

نستخدم sum لحساب المجموع الكلي لقيم عمود ما، وهناك عدة اعتبارات يجب مراعاتها عند

استخدام sum

✓ يستخدم sum مع القيم العددية فقط.

✓ تهمل sum القيمة NULL عند حسابه المجموع الكلي.

✓ تهمل sum القيم المتكررة في العمود عند حسابه المجموع الكلي وذلك إذا صرحتنا

بكتابة DISTINCT قبل إسم العمود.

مثال

يمكنا حساب المجموع الكلي لعمود SAL في جدول EMP، وذلك بكتابة الكود التالي:

```
SQL>SELECT      sum (SAL)
      FROM    emp;
```

SUM(SAL)
29025

2-1-2 استخدام الدالة nvl مع الدوال التجميعية

نستخدم الدالة NVL مع الأعمدة التي تحتوي القيم الفارغة NULL، حيث تقوم بمعالجة مشاكل

القيم الفارغة باستبدالها بالقيمة الصفرية وبالتالي يمكننا استخدام الدوال التجميعية مع الأعمدة التي

تحتوي قيم فارغة لكي تشمل جميع السجلات.

```
SQL> SELECT AVG (NVL(comm, 0) )
  2   FROM    emp;
```

مثال

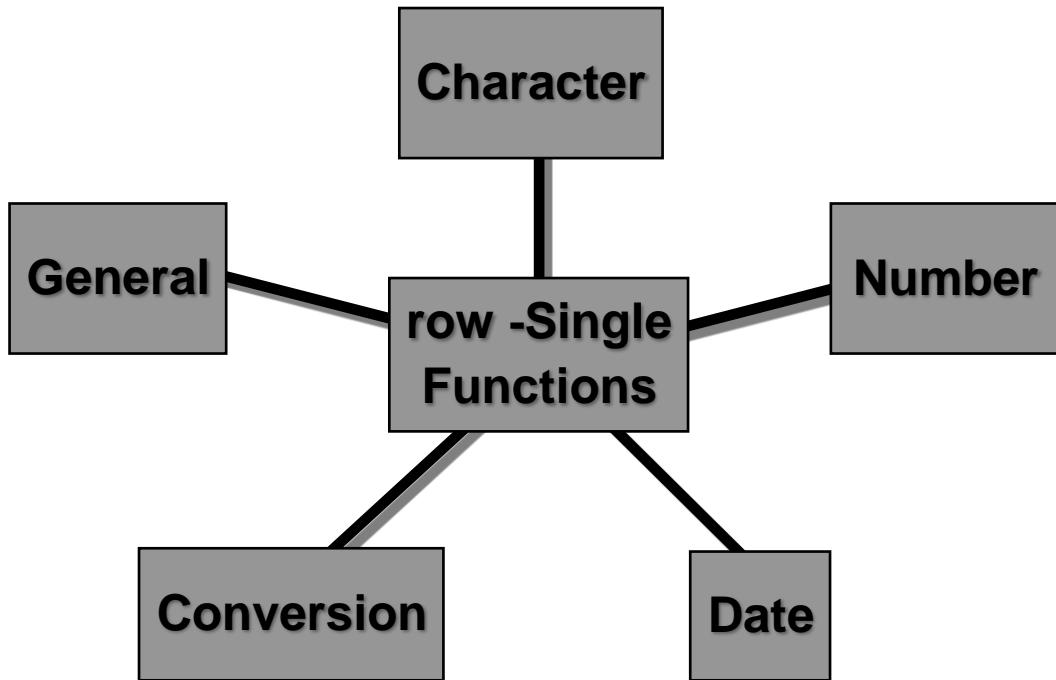
2-2 استخدام single row functions

تعالج single row functions قيم العمود المختار من الجدول وتقدم النتيجة على

شكل قيمة وحيدة لكل سجل ، هذا في حين تستخدم group function في معالجة قيم

العمود المختار من الجدول وتقدم النتيجة على شكل قيمة وحيدة تخص العمود المختار

1-2-2 أنواع single row functions



1-1-2-2 character functions

يمتلك أوراكل عدد من الدوال التي تستخدم لتعديل أنماط البيانات النصية (char,

varchr)، وسوف نتعرض لأهم هذه الدوال النصية في هذا الفصل وهي:

الاستخدام	الدالة النصية
تغيير حالة الحرف الأول من كل سلسلة نصية إلى حرف كبير و تحويل حالة الأحرف التالية للحرف الكبير إلى حالة الأحرف الصغيرة.	INITCAP
يعيد عدد الأحرف المكونة لسلسلة حرفية	LENGTH
تحويل كل حرف ضمن سلسلة حرفية إلى أحرف صغيرة	LOWER
تحويل كل حرف ضمن سلسلة حرفية إلى أحرف كبيرة	UPPER
دمج سلسلة حرفية مع سلسلة حرفية أخرى	CONCAT
استبدال سلسلة حرفية بسلسلة حرفية أخرى	REPLACE
إيجاد الكلمات المتشابهة من حيث اللفظ الصوتي	SOUNDEX

نسخ الجزء المحدد من الكلمة ابتداء من حرف محدد	SUBSTR
يعيد قيمة عدبية تمثل موضع مجموعة أحرف ضمن سلسلة حرفية	INSTR

مثال 1

سنقوم في المثال باستعراض أمثلة على الدوال النصية كما هو موضح بالجدول التالي

الدالة FUNCTION	النتيجة RESULT
LOWER ('JICC Center')	('jicc center')
UPPER ('jicc Center')	('JICC CENTER')
INITCAP ('JICC Center')	('Jicc Center')
CONCAT('JICC'), ('CENTER')	JICCCENTER
SUBSTR('String', 1, 3)	Str
Length('String')	6
INSTR('String', 't')	2

مثال 2

```
SQL>SELECT ename, CONCAT (ename, job),
      LENGTH (ename), INSTR(ename, 'A')
      FROM emp;
```

ENAME	CONCAT(ENAME, JOB)	LENGTH(ENAME)	INSTR(ENAME, 'A')
SMITH	SMITHCLERK	5	0
ALLEN	ALLENSalesman	5	1
WARD	WARDSalesman	4	2
JONES	JONESMANAGER	5	0
MARTIN	MARTINSalesman	6	2
BLAKE	BLAKEMANAGER	5	3
CLARK	CLARKMANAGER	5	3
SCOTT	SCOTTANALYST	5	0
KING	KINGPRESIDENT	4	0
TURNER	TURNERSalesman	6	0
ADAMS	ADAMSCLERK	5	1
JAMES	JAMESCLERK	5	2
FORD	FORDANALYST	4	0
MILLER	MILLERCLERK	6	0

2-1-2 الدوال الرقمية Number functions

يمتلك أوراكل عدد من الدوال الرقمية التي تستخدم مع أنواع البيانات الرقمية Numeric

، وسوف نتعرض لأهم هذه الدوال الرقمية في هذا الفصل وهي:

الاستخدام	الدالة الرقمية
تستخدم لتقريب قيمة رقمية إلى عدد أرقام محدد بعد الفاصلة	ROUND
تستخدم لقطع أو قص القيم إلى دقة محددة	TRUNC
يستخدم لتحديد باقي القسمة الناتج عن عملية ما	MOD

مثال 1

سنقوم في المثال باستعراض أمثلة على الدوال الرقمية كما هو موضح كالتالي

- | | |
|----------------------|-------|
| 1) ROUND (42.926, 2) | 42.93 |
| 2) TRUNC (44.926, 2) | 44.92 |
| 3) MOD (1600, 300) | 100 |

مثال 2

```
SQL> SELECT ROUND(42.923,2),ROUND(43.923,0),
  2 ROUND (44.923,-1)
  3 FROM    DUAL;
```

مثال 3

ROUND(42.923,2)	ROUND(43.923,0)	ROUND(44.923,-1)
42.92	44	40

```
SQL>SELECT    TRUNC(35.832,2),TRUNC(65.323),
  2 TRUNC (65.843,-1)
  3 FROM    DUAL;
```

<code>TRUNC(35.832, 2)</code>	<code>TRUNC(65.323)</code>	<code>TRUNC(65.843, -1)</code>
35.83	65	60

مثال 4

```
SQL>SELECT ename, sal, comm, MOD(sal,comm)
  2 FROM emp;
```

ENAME	SAL	COMM	MOD(SAL, COMM)
SMITH	800		
ALLEN	1600	300	100
WARD	1250	500	250
JONES	2975		
MARTIN	1250	1400	1250
BLAKE	2850		
CLARK	2450		
SCOTT	3000		
KING	5000		
TURNER	1500	0	1500
ADAMS	1100		
JAMES	950		
FORD	3000		
MILLER	1300		
14 row selected			

3-1-2-2 التاريخ ودوال التاريخ Date and Date functions

يمتلك أوراكل عدداً من الدوال التي تستخدم مع أعمدة التاريخ، حيث يمكن إظهار التاريخ بشكل

حرفي، كما يمكننا استخدام التاريخ في عمليات الجمع والطرح، كما يمكن إضافة قيمة ما تاريخ ما

ونحصل بذلك على قيمة تاريخ جديدة، وسوف نتعرض لأهم هذه الدوال وهي موضحة في الجدول

التالي

الاستخدام	الدالة
يحدد عدد الأشهر بين تاريخين	<code>MONTHS_BETWEEN(date1, date2)</code>

يحدد اليوم التالي من الأسبوع الذي يلي التاريخ المحدد	<code>NEXT_DAY(date, 'day')</code>
يستخدم لحساب تاريخ جديد بناء على عدد محدد من الأشهر	<code>ADD_MONTHS(date, numberof months)</code>
يحدد آخر يوم في الشهر الحالي أو لن تاريخ محدد	<code>LAST_DAY(date1)</code>

١، مثلاً

سنقوم في المثال باستعراض أمثلة على الدوال التاريخية كما هو موضح كالتالي

```
MONTHS_BETWEEN ('1-SEP-93', '11-JAN-44')          19.6774194
```

ADD_MONTHS('11-JAN-96', 6) '11-jul-96'

```
NEXT_DAY('01-SEP-95','friday')          '8-sep-  
95'
```

```
LAST_DAY('01-sep-95')          '30-sep-95'
```

```
SQL> SELECT SYSDATE  
      2 FROM dual;
```

۲، ۱۱۸

SYSDATE

20/01/04

دالة SQL Function Sysdate تستخدم لاسترجاع التاريخ والوقت الحاليين.

4-1-2-2 الدوال التحويلية conversion functions

يستخدم أوراكل عدة دوال تحويلية من أهم هذه الدوال التحويلية

✓ `To_number` ويستخدم لتحويل نوع البيانات إلى نوع البيانات الرقمية. فمثلاً يمكننا

استخدام `To_number` لتحويل السلسلة الحرفية السنة إلى حقل رقمي وذلك لاستخدام

أحد الدوال الرقمية على هذا الحقل مثل `MOD`.

✓ `To_char` ويستخدم لتحويل نوع البيانات إلى نوع البيانات النصية، فمثلاً يمكننا

استخدامه لتغيير تاريخ الميلاد للموظف إلى حقل حرفي وذلك لتتناسب ليظهر بتنسيق

خاص ، ويعتبر من أهم الدوال التحويلية في أوراكل، ومن أهم استخداماته هو تحويل نوع

البيانات الرقمية إلى نوع البيانات النصية وذلك لإضفاء عليها تنسيق خاص مثل إظهار

علامة العلامة بجوار الرقم والتحكم في رمز العلامة العشرية وعدد الأرقام بعد العلامة

العشرية وهناك عدة رموز يمكننا استخدامها عند تحويل البيانات الرقمية إلى نوع البيانات

النصية ، كما هو موضح بالجدول التالي :

الرمز	الاستخدام
9	يمثل رقم <code>number</code>
0	يمثل الصفر <code>zero</code>
\$	يمثل علامة الدولار <code>dollar sign</code>
L	يمثل علامة العملة علي النظام <code>local currency symbol</code>
.	يمثل رمز العلامة العشرية <code>Decimal point</code>
,	رمز تجميع الأرقام <code>A thousand indicator</code>

✓ `To_date` ويستخدم لتحويل نوع البيانات إلى نوع البيانات التاريخية.

مثال 1:

في هذا المثال سنقوم بتحويل نوع بيانات عمود `hiredate` من نوع `date` إلى نوع البيانات الحرفية وذلك بإستخدام `TO_CHAR`

```
SQL>SELECT ename, TO_CHAR(hiredate, 'fmDD Month YYYY') HIREDATE
  2 FROM      emp;
```

ENAME	HIREDATE
SMITH	كانون الأول 17 1980
ALLEN	شباط 28 1981
WARD	شباط 22 1981
JONES	نisan 2 1981
MARTIN	أيلول 28 1981
BLAKE	أيار 1 1981
CLARK	حزيران 9 1981
SCOTT	نisan 19 1987
KING	تشرين الثاني 17 1981
TURNER	أيلول 8 1981
ADAMS	أيار 23 1987
JAMES	كانون الأول 3 1981
FORD	كانون الأول 3 1981
MILLER	كانون الثاني 23 1982

تم اختيار 14 صف

SQL>

تلميح

- ✓ تنسيق التاريخ لابد أن يوضع بين علامتي تنسيق مفردة ' ' .

- ✓ نفصل بين التاريخ والتنسيق بفاصلة

مثال 2

في هذا المثال سنقوم بتحويل نوع بيانات عمود sal من نوع number إلى نوع البيانات

الحرفية وذلك بإستخدام `TO_CHAR` وذلك لعمل تنسيق خاص لحقل المرتب مع استخدام الرموز

```
SQL>SELECT TO_CHAR(sal, '$99,999') SALARY
  2 FROM emp;
```

SALARY
\$800
\$1,600
\$1,250
\$2,975
\$1,250
\$2,850
\$2,450
\$3,000
\$5,000
\$1,500
\$1,100
\$950
\$3,000
\$1,300

14 row selected

5 الدوال العامة General functions

من أهم الدوال العامة في SQL دالتي decode,nvl single row function وفيما يلي توضيح لاستخدامات كلتا الدالتين في SQL.

1-2-2 الدالة NVL

تستخدم الدالة NVL لاستبدال القيمة الفارغة بقيمة أخرى، ويمكن استخدام NVL مع نوع البيانات الرقمية وغير الرقمية، ويعتبر من أهم استخدامات تلك الدالة هو التغلب على مشكلة القيم الصفرية في البيانات الرقمية إذا تقوم NVL باستبدال القيم الفارغة بقيم صفرية، وهناك عدة اعتبارات يجب مراعاتها عند استخدام NVL

✓ تستخدم الدالة NVL في تحويل القيم الفارغة إلى قيم فعلية.

✓ يمكنها معالجة أنواع البيانات الرقمية NUMERIC DATA، والنصية

.DATE، CHARACTER STRING

✓ عند تحويل القيم الفارغة عليك بالالتزام بنفس نوع البيانات Data types في

العمود الذي يحتوي القيم الفارغة فعلى سبيل المثال:

- Nvl(comm, 0)
- Nvl(job, 'no job yet')

- `Nvl(hiredate, '01-jan-98')`

مثال

ENAME	SAL	COMM (SAL*12)+NVL(COMM,0)
SQL> SELECT ename,sal,comm, (sal*12) +NVL(comm,0)		
A 2 FROM emp;		
W		
J		
M		
BLAKE	2850	34200
CLARK	2450	29400
SCOTT	3000	36000
KING	5000	60000
TURNER	1500	18000
ADAMS	1100	13200
JAMES	950	11400
FORD	3000	36000
MILLER	1300	15600
14 row selected		

Decode الدالة 2-5-1-2-2

تسهل الدالة Decode إجراء الاستعلام الشرطي في SQL، وتماثل عمل كلاً من CASE،

Decode ، والصيغة العامة لاستخدام الدالة IF-THEN-ELSE

`DECODE (col/expression, search1, result1
[, search2, result2,...,] [, default])`

في المثال التالي نستخدم Decode لإجراء الاستعلام الشرطي التالي، فإذا كانت الوظيفة ANALYST يتم حساب الحوافز على أساس 5% من المرتب، وإذا كانت الوظيفة SALESMAN يتم حساب الحوافز على أساس 6% من المرتب، وإذا كانت الوظيفة MANAGER يتم حساب الحوافز على أساس 25% من المرتب، فيما عدا ذلك لا يتم احتساب حوافز.

```
SQL> SELECT job,sal,
  2 DECODE(job,'ANALYST',SAL*.05
  3           'salesman',    SAL*.06
  4           'MANAGER',   SAL*.25,
  5           SAL)
  6           BONUS
  7 FROM emp;
```

JOB	SAL	BONUS
CLERK	800	800
SALESMAN	1600	1600
SALESMAN	1250	1250
MANAGER	2975	743.75
SALESMAN	1250	1250
MANAGER	2850	712.5
MANAGER	2450	612.5
ANALYST	3000	150
PRESIDENT	5000	5000
SALESMAN	1500	1500
CLERK	1100	1100
CLERK	950	950
ANALYST	3000	150
CLERK	1300	1300

14 row selected

استخدام GROUP BY - 3

كنا فيما سبق كانت SQL تعالج الجدول أو الجداول بالكامل أو السجلات المحققة لشرط معين في جملة WHERE، إلا أننا في بعض الأحيان نريد إجراء عمليات منطقية على سجلات معينة مثل إظهار القيمة العظمى والدنيا والمتوسط الحسابي لكل إدارة من الإدارات على حدة، لإجراء مثل هذا نحتاج إلى استخدام group by Group، حيث تستخدم جملة group by لتحديد الأعمدة التي يتم التجميع بناء عليها أو عندما نريد تطبيق الدوال التجميعية مثل avg، sum، count، max، min على سجلات من الجدول معزولة على شكل مجموعات فرعية، حيث تميز كل مجموعة بنفس قيم المعطيات، بحيث تقسم group by سجلات group functions إلى مجموعات صغيرة متشابهة، ثم يمكنك استخدام الجدول إلى مجموعات تلخيصية لكل مجموعة من هذه المجموعات على حدة، والصيغة العامة لاسترجاع معلومات تلخيصية لكل مجموعة من هذه المجموعات على حدة، والصيغة العامة لاستخدام group by تأخذ الشكل التالي:

```
SQL>SELECT column, group_function(column)
      FROM table
      [WHERE condition]
      [GROUP BY group_by_expression]
      [ORDER BY column];
```



✓ تقع جملة group by بعد جملة where أو بعد جملة From في حالة عدم

وجود جملة where.

✓ تقسم group by سجلات الجدول إلى مجموعات صغيرة متشابهة، ثم يمكنك

استخدام group functions لاسترجاع معلومات تلخيصية لكل مجموعة من

هذه المجموعات على حدة.

✓ تتعامل GROUP BY مع القيم الفارغة NULL كمجموعة مميزة عن غيرها.

✓ تمكنا بإنشاء مجموعات فرعية ضمن مجموعات فرعية أخرى مثل:

GROUP BY deptno, sal

وهناك عدة اعتبارات يجب مراعاتها عند استخدام Group by

✓ يمكن أن تحتوي عدة أعمدة وتعابير.

✓ يتم وضع فواصل بين أسماء الأعمدة.

✓ يجب أن تحتوي على كل الأعمدة المكتوبة بجملة Select والتي لا تستخدمها

دوال التجميع.

مثال 1

في هذا المثال نريد حساب المجموع الكلي لمرتبات كل إدارة علي حدة، لفعل ذلك يجب علينا

استخدام عبارة group by في جملة select

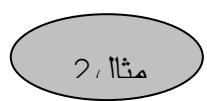
```
SQL> SELECT deptno, sum(sal)
  2 FROM emp
  3 GROUP BY deptno;
```

DEPTNO	SUM(SAL)
10	8750
20	10875
30	9400



تلميح

- ✓ يجب أن تحتوي جملة group by على كل الأعمدة المكتوبة بجملة والتي لا تستخدمها دوال التجميع(وهو في المثال السابق حقل deptno الذي يراد يتم التجميع بناء عليه).



مثال ٢

يمكننا إجراء المثال السابق و حساب المجموع الكلي لمرتبات كل إدارة علي حدة

باستخدام عمود deptno في جملة select بدون كتابة عمود deptno

```
SQL> SELECT sum(sal)
  2 FROM emp
  3 GROUP BY deptno;
```

SUM(SAL)

8750
10875
9400



تلميح

- ✓ يمكننا إنشاء مجموعات من السجلات باستخدام group by بناء علي أعمدة ليست موجودة بجملة Select

3، ١اً مثا

في هذا المثال سنوضح إمكانية قيام GROUP BY بإنشاء مجموعات فرعية ضمن مجموعات فرعية أخرى

```
SQL>SELECT deptno, job, sum(sal)
  2 FROM emp
  3 GROUP BY deptno, job;
```

تلميح

✓ أولاً يتم إنشاء مجموعات متشابهة بناء على رقم الإدارة deptno

(First, the rows are grouped by department number)

✓ ثانياً في مجموعات الإدارة يتم عمل مجموعات متشابهة من للسجلات بناء على الوظيفة.

(Second, within the department number groups, the rows are grouped by job title.)

DEPTNO	JOB	SUM(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	6000
20	CLERK	1900
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	5600
9 row selected		

4، ١اً مثا

عند إجراء المثال التالي تظهر لنا رسالة الخطأ التالية(ORA-00934) : وظيفة المجموعة غير مسموحة هنا) والتي تعني أننا لا نستطيع استخدام الدوال التجميعية Group Function مع

Where جملة

```
SQL> SELECT deptno, MAX(sal)
  2 FROM emp
  3 WHERE MAX (sal) > 2000
  4 GROUP BY deptno;
```

```
WHERE MAX (sa1) > 3000
*
ERROR at line 3:
ORA-00934: group function is not allowed here
```

- استخدام Having (قييد الاختيار ضمن المجموعة)

(Restriction group selection)

قد لا نريد استرجاع دائمًا جميع المجموعات الفرعية المختلفة في الجدول ضمن استعلام واحد،

فقدم لنا SQL جملة Having لحذف المجموعات الفرعية من التي لا نريد استرجاعها في الاستعلام أو التقرير، وتعتبر جملة Having مشابهة لجملة Where من حيث أنها تحدد فيما

إذا كانت السجلات قابلة لل اختيار أو لا، وتحتوي جملة Having على محدودات تستخدم لتقدير

السجلات، حيث يمكننا وضع عدة شروط في جملة Having، كما يمكننا استخدام المعاملات

المنطقية معها مثل or , and ، وقد يتadar إلى الذهن لماذا لا نستخدم عوضا عنها جملة

، لا يمكننا ذلك حيث يكمن الاختلاف بين جملة Having وجملة Where في أن

محددات الجملة Having يجب أن تكون تابع تجميع، ويتم استخدام Having فقط عند يتم حساب قيمة مجمعة بواسطة Select، بينما عمل Where هو ترشيح السجلات التي سيبطبق عليها لاحقاً جملة Group by، وليس ترشيح المجموعات، والصيغة العامة لاستخدام having موضحة كالتالي

```
SQL>SELECT column, group_function(column)
      FROM table
      [WHERE condition]
      [GROUP BY group_by_expression]
      [HAVING group_condition]
      [ORDER BY column];
```

وهناك عدة اعتبارات يجب مراعاتها عند استخدام HAVING

- ✓ تقع جملة Having بعد فقرة Group by
- ✓ تستخدم Having فقط مع Group by ولا يمكن استخدامها بدونها.
- ✓ نستخدم Having مع الدوال التجميعية التالية فقط (MAX, MIN, AVG, COUNT, SUM)
- ✓ يمكن تحديد عدة شروط في جملة Having وذلك باستخدام المعاملات المنطقية .and, or
- ✓ يمكننا استخدام كلًّا من Where,Having في جملة Select

مثال:

```
SQL>SELECT job, SUM(sal) PAYROLL
  2 FROM      emp
  3 WHERE      job NOT LIKE 'SALES%'
  4 GROUP BY job
  5 HAVING     SUM (sal)<6000
  6 ORDER BY SUM (sal);
```

في المثال السابق نقوم باسترجاع الوظيفة والمجموع الكلي للمرتبات الشهرية لكل وظيفة على حده بشرط أن يكون المجموع الكلي للمرتبات الشهرية لكل وظيفة أقل من 6000 ريال باستثناء وظيفة salesman، ثم الترتيب التصاعدي بناء على المجموع الكلي للمرتبات الشهرية.

JOB	PAYROLL
CLERK	4150
PRESIDENT	5000

5- استخدام وعمل استعلام فرعي(Sub query)

تمكننا من جعل الاستعلامات متداخلة مع بعضها البعض بشكل نموذجي، بحيث ينتج عن الاستعلام الداخلي (inner query) قيمًا تتحقق في قسم الشرط للاستعلام الخارجي (outer query) لمعرفة فيما إذا كان الشرط سيتحقق عندها أم لا

5-1 كيف تعمل الاستعلامات الفرعية

نفترض أنك تريد كتابة استعلاماً لاسترجاع من يأخذ مرتبًا أكثر من محمد؟ نجد أننا نحتاج إلى استعلامين

الأول: ما هو مرتب محمد؟

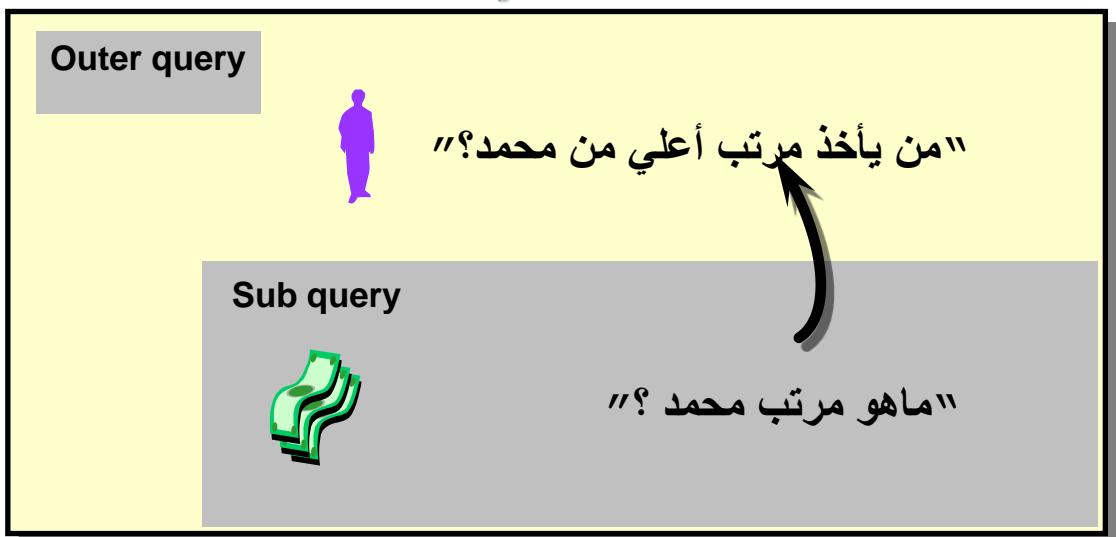
الثاني: البحث عن من يأخذ مرتبًا أعلى من محمد؟

و لحل مثل تلك المشكلة لابد من دمج الاستعلامات معاً بمعنى استخدام استعلام داخل استعلام

أخر، حيث يعيد الاستعلام الداخلي أو الفرعى يعيد قيمة يستخدمها الاستعلام الرئيسي أو الخارجى

لمعرفة فيما إذا كان سيتحقق الشرط عندها أم لا

من يأخذ مرتبًا أعلى من محمد



للمزيد

- ✓ ينفذ الاستعلام الفرعى أو الداخلى أولاً ثم ينفذ الاستعلام الرئيسي أو الخارجى.
- ✓ يولد أو يعيد الاستعلام الداخلى نتيجة يستخدمها الاستعلام الخارجى لمعرفة ما إذا سيتحقق الشرط عنها أم لا.
- ✓ الصيغة العامة لعمل واستخدام الاستعلامات الفرعية Sub queries كالتالى

```
SQL>SELECT select_list  
FROM table  
WHERE expr operator  
      (SELECT select_list  
           FROM      table);
```



مثال

```
SQL> SELECT ename
  2  FROM emp          5000
  3 WHERE sal >      ←
  4       (SELECT sal
  5        FROM emp
  6        WHERE empno=1);
```

في المثال السابق نجد أن الاستعلام الداخلي يحدد مرتب الموظف الذي رقمه هو 1،
يأخذ الاستعلام الخارجي أو الرئيسي هذه النتيجة ويستخدمها لاسترجاع الموظفين الذين يتلقاًون
مرتبًا أعلى من هذا المرتب

5-2 أنواع الاستعلامات الفرعية

يوجد لدينا نوعين من الاستعلامات الفرعية وهي كالتالي:

5-2-1 الاستعلامات الفرعية التي تعيد سجل واحد

(single-row sub queries)

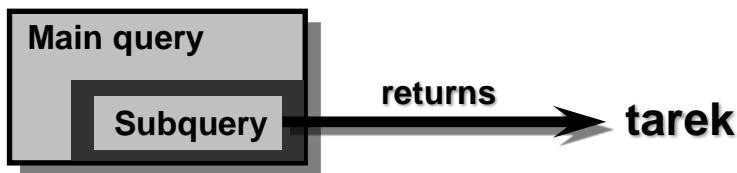
5-2-2 الاستعلامات الفرعية التي تعيد أكثر من سجل واحد

(Multiple-row sub queries)

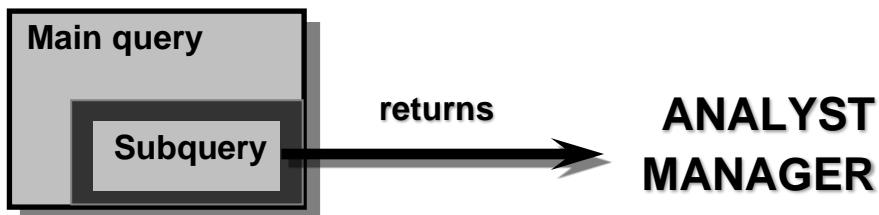
5-2-3 الاستعلامات الفرعية التي تعيد أكثر من عمود واحد

(Multiple-column sub queries)

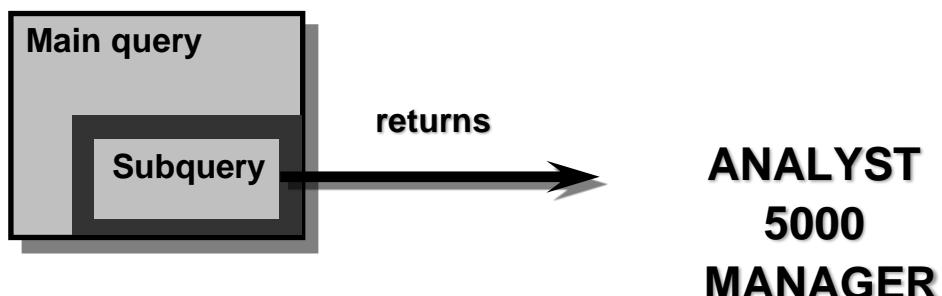
Single-row subquery •



• Multiple-row subquery



• Multiple-column subquery



5-2-1 الاستعلامات الفرعية التي تعيد سجل واحد

كما سبق وتعلمنا أن (single-row sub queries) تعيد سجل واحد فقط ،

ويستخدم ذلك النوع من الاستعلامات مع عوامل المقارنة الموضحة بالجدول التالي :

Operator المعامل	meaning المعنى
=	يساوي
>	أكبر من
>=	أكبر من أو يساوي

<	أقل من
<=	أقل من أو يساوي
<>Or ! =	لا يساوي

مثال 1

في هذا المثال سنقوم باسترجاع بيانات الموظفين (الاسم، الوظيفة) للموظفين الذين لهم نفس

الوظيفة للموظف رقم 555

```
SQL>SELECT ename, job
  FROM emp
 WHERE job=
       (SELECT job
        FROM emp
       Where empno=555) ;
```

ENAME	JOB
SMITH	CLERK
ADAMS	CLERK
JAMES	CLERK
MILLER	CLERK

مثال 2

في هذا المثال سنقوم باسترجاع بيانات الموظفين (الاسم، الوظيفة) للموظفين الذين لهم نفس

الوظيفة للموظف رقم 7698 ومرتباتهم أكبر من راتب الموظف رقم 7934

```

SQL> SELECT      ename, job
  2  FROM        emp
  3 WHERE       job =
  4          (SELECT    job
  5            FROM      emp
  6            WHERE     emplno = 7698)
  7 AND        sal >      1300
  8          (SELECT    sal
  9            FROM emp
 10           WHERE    emplno = 7934);

```

ENAME	JOB
JONES	MANAGER
BLAKE	MANAGER
CLARK	MANAGER

تمييز

- ✓ لابد من وضع الاستعلام الفرعى بين قوسين.
- ✓ لا تستخدم جملة Order By داخل الاستعلام الفرعى.

5-2-1 استخدام الدوال التجميعية في Sub query

تتميز الدوال التجميعية بأنها تولد أو تعيد قيمة وحيدة مهما كان عدد السجلات المختارة، وينبغي

علينا الانتباه إلى أن الدوال التجميعية التي هي دوال تجميع معرفة في جملة Group By ستولد

وتعيد عدة قيم، ولهذا لا يسمح باستخدامها مع هذا النوع من الاستعلامات الفرعية التي تعيد سجل

واحد، حتى لو استخدامنا HavingGroup by لتوليد مجموعة وحيدة لخرج الاستعلام الفرعى، ولهذا يجب علينا استخدام الدوال التجميعية المرغوبة في جملة where التي تحذف المجموعات غير المرغوبة.

في هذا المثال سنقوم باسترجاع بيانات الموظف (الاسم، الوظيفة، المرتب) الذى يتلقى أعلى مرتب، حيث يمكننا تنفيذ هذا الاستعلام باستخدام group function داخل الاستعلام الفرعى كما يلى:

```
SQL> SELECT ename, job, sal
  2 FROM emp
  3 WHERE      sal =
  4           (SELECT MAX(sal)
  5            FROM      emp);
```

ENAME	JOB	SAL
KING	PRESIDENT	5000

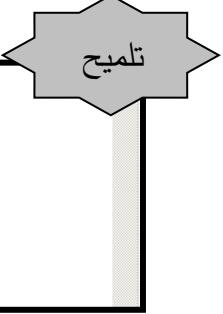
2-1-2-5 استخدام Sub query مع (having و group by)

يمكن للاستعلامات الفرعية أن تستخدم في جملة having، ويمكنها استخدام الدوال التجميعية

أو تستخدم group having كما بالمثال التالي:

في هذا المثال نريد إنشاء مجموعات أصغر مرتبات بناء على عمود رقم الإدارة ، مع تقييد ذلك بشرط وهو أن يكون أقل مرتب في تلك الإدارات أكبر من أقل مرتب في الإدارة رقم 10

```
SQL> SELECT deptno, MIN(sal)
  2 FROM emp
  3 GROUP BY deptno
  4 HAVING MIN(sal) >
  5       (SELECT MIN(sal)
  6        FROM emp
  7        WHERE deptno = 10);
```



تلميح

DEPTNO	MIN(SAL)
10	1300
30	950

✓ يجب علينا التأكد عند استخدام الاستعلامات الفرعية في القسم الشرطي (الذي فيه عامل المساواة أو عدم المساواة) من أن الاستعلام الفرعي سيعيد سجلاً واحداً فقط، وإلا سيصدر خطأ إذا أعاد أكثر من قيمة واحدة، ولن يفشل الأمر إذا لم يعيد الاستعلام الفرعي أية قيمة ولكننا عندها لن نحصل على خرج من الاستعلام الفرعي.

5-2-2 الاستعلامات الفرعية التي تعيد أكثر من سجل واحد

تعيد الاستعلامات الفرعية متعددة السجلات أكثر من سجل، وتنطلب هذه الأنواع من الاستعلامات الفرعية معالجاً يمكن استخدامه لتقييم عدة قيم، فعلى سبيل المثال لا الحصر يمكننا استخدام

المعامل IN مع الاستعلامات الفرعية متعددة الأسطر لأنه يقيّم مصفوفة من القيم، والجدول التالي يوضح المعاملات التي يمكننا استخدامها لتقديم الاستعلامات متعددة السجلات.

المعامل Operator	الوصف
IN	يكون المحدد مساوياً لأي من القيم التي يعيدها الإستعلام الفرعى لهذا الشرط حتى يصبح محققا (True)
ANY	يقارن المحدد بكل قيمة يعيدها الإستعلام الفرعى ، ويكون الشرط محققا (True) إذا كان أي عنصر من المجموعة يحقق الشرط
ALL	يقارن المحدد بكل قيمة يعيدها الإستعلام الفرعى ، ويكون الشرط محققاً

في هذا المثال نريد استرجاع بيانات الموظفين (رقم الموظف، الاسم، الوظيفة) ذوي المرتبات أقل من أي clerk والذين لا يعملون في وظيفة clerk

مثال

```
SQL> SELECT empno, ename, job
  2 FROM emp
  3 WHERE sal < ANY(
  4   (SELECT sal
  5    FROM emp
  6    WHERE job = ' CLERK ')
  7 AND job <> 'CLERK';
```

any < : يعني أقل من minimum ✓

للمزيد

maximum < any ✓
IN = any ✓

5-2-3 الاستعلامات الفرعية التي تعدد أكثر من عمود واحد

(Multiple-column sub queries)

رأينا في الأمثلة السابقة استعلامات تستخدم أعمدة وحيدة كمحددات تقييم، وأحياناً لتنفيذ نوع من الشروط يجب علينا استعراض عدة أعمدة وكأنها محدد واحد، ويجب إحاطة تلك الأعمدة بين قوسين ووضع فواصل بين الأعمدة.



مثال

```
SQL> SELECT ename, deptno, sal, comm
  2 FROM emp
  3 WHERE (sal, NVL (comm, 0)) IN
  4          (SELECT sal, NVL (comm, 0)
  5           FROM   emp
  6           WHERE deptno = 30);
```

في المثال السابق نريد استرجاع بيانات الموظف(الاسم، رقم الإدارة، المرتب، الحوافز)، لأي موظف يتماثل في المرتب والحوافز مع أي موظف في الإداره رقم 30 .

ENAME	DEPTNO	SAL	COMM
JAMES	30	950	
WARD	30	1250	500
MARTIN	30	1250	1400
TURNER	30	1500	0
ALLEN	30	1600	300
BLAKE	30	2850	

6 row selected

الأسئلة

المجموعة الأولى:

ضع علامة (✓) أمام العبارة الصحيحة وعلامة (✗) أمام العبارة الخاطئة مع تصحيح الخطأ.

1) ينفذ الاستعلام الفرعي أو الداخلي أولاً ثم ينفذ الاستعلام الرئيسي أو الخارجي

2) في الرابطة المتكافئة لابد أن كل سجل في الجدول الرئيس الأب يقابله سجل أو

سجلات في الجدول المرتبط به الابن.

3) الرابطة غير المتكافئة تمتلك الجداول أعمدة وقيمًا مشتركة.

4) في الرابطة الخارجية تكون السجلات المرتبطة يمكن أن تحتوي قيم مشتركة ويمكن

أن لا تحتوي.

(5) الروابط الذاتية Self join تمكنا من عمل رابطة ذاتية عند ربط الجدول مع نفسه.

(6) يطلق على العلاقة inner join أو simple join أو equijoin.

(7) في جملة Where يمكننا استخدام TABLE Aliases.

(8) تقوم الدوال التجميعية Group Functions بمعالجة قيم العمود المختار

من الجدول وتقدم النتيجة في شكل قيمة وحيدة تخص العمود المختار.

Select MAX(avg(sal)) from emp; (9)

(10) يقوم بحساب عدد السجلات بما فيها السجلات التي تحتوي

على قيمة فارغة.

(11) يمكن لجميع الأعمدة في جملة select أن تكون كلها دوال تجميعية.

(12) يمكن استخدام الدوال التجميعية ضمن التعبير الحسابية.

(13) يستخدم AVG مع القيم العددية فقط ويهمل NULL عند حسابه

للقيمة الوسطي.

(14) To_number ويستخدم لتحويل نوع البيانات إلى نوع البيانات الحرفية.

(15) To_char ويستخدم لتحويل نوع البيانات الحرفية إلى نوع البيانات الرقمية.

المجموعة الثانية:

أسئلة لتدريب الطالب على أسئلة الاختيار المتعدد:

(1) من الأسباب التي تدعونا إلى إنشاء العلاقات

(a) دمج الأعمدة من جدولين أو أكثر.

(b) إختيار أعمدة موجودة في جدول واحد بناء على شرط يطبق على عمود آخر.

.1 & 2 (c)

(d) ليس شيء مما سبق.

2) عند استرجاع بيانات من جدولين أو أكثر يتم إضافة شرط الربط في عبارة `Where`.

(a) العبارة صحيحة.

(b) العبارة خاطئة.

3) من أنواع العلاقات بين الجداول

(a) الرابطة المتكافئة.

(b) الرابطة الذاتية.

(c) الرابطة الخارجية .

(d) جميع ما سبق.

4) في الرابطة الخارجية `Outer join` ::

(a) تمتلك الجداول أعمدة مشتركة.

(b) السجلات يمكن أن تحتوي قيم مشتركة أولاً تحتوي.

2&1 (c)

(d) ليس شيء مما سبق .

5) الرابطة الديكارتية تقوم ::

(a) بحذف شرط الربط بين الجدولين في جملة `Where`

(b) يقوم أوراكل بربط كل سجل في الجدول الأول مع كل سجل في الجدول الثاني.

(c) يقوم أوراكل بإنشاء ومعالجة كمية كبيرة جداً من البيانات

(d) جميع ما سبق

6) هناك عدة اعتبارات يجب إتباعها عند استخدام **group functions** من:

- (a) وضع إسم العمود بين قوسين () بعد الدالة مباشرة
- (b) يمكن لجميع الأعمدة في جملة Select أن تكون كلها دوال تجميعية
- (c) استخدام دالة تجميعية أخرى غير مسموح به في SQL
- (d) الدوال التجميعية تتجاهل القيم الفارغة NULLVALUES
- (e) يمكن استخدام الدوال التجميعية ضمن التعبير الحسابية
- (f) جميع ما سبق

7) الدالة :**MAX**

- (a) يمكن استخدام max مع أي نوع من البيانات.
- (b) تهمل الدالة MAX القيم NULL.
- (c) 2&1
- (d) يمكن استخدام max مع نوع البيانات الحرفية فقط.

8) الدالة :**AVG**

- (a) يستخدم AVG مع القيم العددية فقط.
- (b) يهمل AVG القيمة NULL عند حسابه للقيمة الوسطي.
- (c) 2&1
- (d) يستخدم AVG مع أي نوع من البيانات.

9) الدالة :**MONTH_BETWEEN**

- (a) تحدد عدد الأشهر بين تاريخين.
- (b) يحدد اليوم التالي من الشهر القادم الذي يلي التاريخ المحدد.

(c) يحدد تاريخ جديد بناء على عدد محدد من الأشهر

2&1 (d)

:LAST_DAY(10) الدالة

(a) تحدد عدد الأشهر بين تاريفين.

(b) يحدد اليوم التالي من الشهر القادم الذي يلي التاريخ المحدد.

(c) يحدد تاريخ جديد بناء على عدد محدد من الأشهر

(d) يحدد آخر يوم في الشهر الحالي أو التاريخ المحدد.

11) هناك عدة اعتبارات يجب مراعاتها عن استخدام الدالة :nvl

(a) تستخدم الدالة NVL في تحويل القيم الفارغة إلى قيم فعلية.

(b) يمكنها معالجة أنواع البيانات الرقمية NUMERIC DATA، والنصية

.CHARACTER STRING، والتاريخية DATE.

(c) عند تحويل القيم الفارغة عليك بالالتزام بنفس نوع البيانات Data types في

العمود الذي يحتوي القيم الفارغة.

A&b (d)

A&b&c (e)

:group by(12

(a) تتعامل GROUP BY مع القيم الفارغة NULL كمجموعة مميزة عن غيرها.

(b) تمكنا بإنشاء مجموعات فرعية ضمن مجموعات فرعية أخرى GROUP BY

A&b (c

(d) ليس شيء مما سبق.

:HAVING(13

(a) نستخدم Having مع الدوال التجميعية التالية فقط

.Select WhereHaving في جملة (b) يمكننا استخدام كلاً من

.Group by Having بعد فقرة (c) تقع جملة

A&B&C (d

A&C (e

:HAVING(14

(f) نستخدم Having مع الدوال التجميعية التالية فقط

.Select WhereHaving في جملة (g) يمكننا استخدام كلاً من

.Group by Having بعد فقرة (h) تقع جملة

A&B&C (i

A&C (j

الفصل الثالث

معالجة البيانات باستخدام

SQL

الأهداف:

إعطاء الطالب صورة موضحة عن كيفية إضافة سجلات جديدة إلى الجداول، وكيفية حذف وتعديل بيانات موجودة في قاعدة البيانات، وكيفية إنشاء وإضافة محددات من نوع

Primary key, foreign key

المحتويات:

1. مقدمة
2. كيفية إضافة سجلات جديدة للجداول.
3. كيفية حذف بيانات موجودة في قاعدة البيانات .
4. كيفية تعديل البيانات الموجودة في قاعدة البيانات .
5. كيفية إنشاء جدول وإضافة محددات من نوع
(primary key, foreign key)

ماذا سنتعلم في هذا الفصل :

في نهاية هذا الفصل يكون الطالب قد اكتسب المهارات والمعرفات التالية :

1. إنشاء وتعديل وإسقاط الجداول
2. كيفية إضافة سجلات جديدة للجداول.
3. كيفية حذف بيانات موجودة في قاعدة البيانات .
4. كيفية تعديل البيانات الموجودة في قاعدة البيانات .
5. كيفية إنشاء جدول وإضافة محددات من نوع
(primary key, foreign key)

مقدمة

كما تعلمنا سابقاً أن عبارات SQL تت分成 إلى ثلاثة فئات رئيسية وهي كالتالي:

1- لغة تعريف البيانات DDL

هي مجموعة من أوامر SQL تستطيع من خلالها إنشاء وتعريف الكائنات في قاعدة البيانات، وتقوم هذه الأوامر بإنشاء أو إسقاط أو تغيير كائن قاعدة البيانات، وهي اختصاراً Create, Alter, Drop, Rename, Truncate للكلمات (Data Definition Language).

2- لغة معالجة البيانات DML

هي مجموعة من العبارات التي تستطيع من خلالها معالجة البيانات الموجودة في قاعدة البيانات من إدراج البيانات وتحديثها أو تحديد أو حذف البيانات، وهي اختصاراً Insert, Update, Delete للكلمات (Data Manipulation Language)، ومن أهم أوامرها

3- لغة التحكم والصلاحيات DCL

هي مجموعة من أوامر SQL تستطيع من خلالها منح أو سحب إمتيازات استخدام كائن قاعدة البيانات، وهي اختصاراً للكلمات (Data Control Language)، ومن أهم أوامرها Grant, Revoke.

سنتعلم في هذا الفصل كيفية التعامل مع البيانات المخزنة في قاعدة البيانات من إضافة سجلات أو حذف سجلات أو تغيير قيمة مخزنة في سجل، بالإضافة إلى كيفية إنشاء الجداول وإضافة محددات من نوع primary key, foreign key

١- كيفية إضافة سجلات جديدة

تمتلك معظم برامج أنظمة قواعد البيانات أدوات ممتازة لإدخال وتعديل وحذف البيانات من الجداول، وهي بلا شك أفضل وأسرع بكثير من إدخال البيانات بعبارات SQL، إلا أن ذلك يمكن المبرمج من معالجة البيانات من ضمن برامجه التي يكتبها، ونستطيع إضافة سجلات جديدة إلى الجدول باستخدام عبارة `insert`، ويوجد لدينا صيغتين من عبارة `insert`، الأولى تسمح بإضافة سجل واحد في نفس الوقت إلى الجدول، والثانية تسمح باختيار مجموعة سجلات من جداول وإدراجها في جدول آخر.

١-١ إضافة سجل واحد إلى الجدول

كما وضمنا سابقاً أن عبارة `insert` لها صيغتين الأولى والتي نحن بصددتها تمكناً من إضافة سجل واحد إلى الجدول في نفس الوقت، والصيغة العامة لها تأخذ الشكل التالي:

```
INSERT INTO table [(column [, column...])]  
VALUES      (value [, value...]);
```

مما سبق نجد أن الصيغة العامة لعبارة `insert` تتتألف مما يلي:

sql هي كلمات محوزة في `insert, into, values`. 1

Table . 2 ويشير إلى إسم جدول وحيد في قاعدة البيانات ، وبالتحديد الجدول الذي نريد

إدراج السجلات به.

Column . 3: وتشير إلى أعمدة الجدول.

4. لا ظهرة من القيم محاطة بـ `()` تقوم sql بإدراجها في الأعمدة المقابلة لها على الترتيب.

مثال 1

لإدراج سجل من البيانات إلى جدول dept، نقوم بكتابة الكود التالي:

```
SQL> INSERT INTO dept (deptno, dname, loc)
  2 VALUES          (50, 'DEVELOPMENT', 'cairo');
1 row created.
```

مثال 2

لإدراج سجل من البيانات إلى جدول emp، نقوم بكتابة الكود التالي:

```
SQL> INSERT INTO EMP (empno, ename, job,
  2           mgr, hiredate, sal, comm,
  3           deptno)
  4 VALUES      (7369, 'ayman', 'programmer',
  5             7372, SYSDATE, 4000, NULL,
  6             50);
1 row created.
```

تلميح

✓ يجب أن تتطابق الأعمدة والقيم المقابلة لها في نوع البيانات .data types.

✓ لائحة القيم لابد أن تأخذ نفس ترتيب الأعمدة في الجدول المراد إضافة السجلات

إليه.

✓ لابد من وجود علاقة واحد إلى واحد بين الأعمدة ولائحة القيم، بمعنى أن توجد قيمة

واحدة من أجل كل عمود وبنفس الترتيب.

✓ قد نجهل قيمة أحد الحقول، وذلك بكتابة كلمة NULL، بشرط ألا يكون العمود

المقابل للقيمة معرفاً بالخاصية NOT NULL.

✓ قد تسبب INSERT مشكلة في التجانس والتكامل المرجعي (وسيتم تناول ذلك عند

إنشاء العلاقات بين الجداول لاحقاً في هذا الفصل).

✓ لابد من وضع الأحرف النصية والتاريخ بين علامتي تصيص مفردة ' ' .

✓ للتأكد من تنفيذ العبارة السابقة وإضافة السجل إلى جدول emp قم بتنفيذ العبارة

التالية:

```
SQL>Select *      From emp;
```

1-2 إضافة عدة سجلات إلى الجدول:

نلاحظ أنه ليس من المنطقي أن تقوم بإدخال 20,000 سجلاً مثلاً في أحد الجداول سجلاً

سجلاً باستخدام الصيغة الأولى من عبارة insert، وإضافة عدة سجلات في نفس الوقت

تمكننا الصيغة الثانية من عبارة insert بإضافة عدة سجلات إلى جدولنا مختارة من

جدول آخر ، والصيغة العامة لها هي:

```
INSERT INTO table([(column [, column...])])  
Select ([(column [, column...])])  
[where conditions];
```

ما سبق نجد أن الصيغة العامة لعبارة insert تتالف مما يلي:

Sql هي كلمات محجوزة في insert, into .1

Table .2 ويشير إلى إسم جدول وحيد في قاعدة البيانات ، وبالتحديد الجدول الذي نريد

إدراج السجلات به.

Column .3 : وتشير إلى أعمدة الجدول.

4. عبارة select هي عبارة عاديّة تستخدم لانتخاب الأعمدة التي ستتسلخ

محتوياتها إلى الجدول المراد إضافة السجلات إليه ، فعلى سبيل المثال نريد إضافة سجلات أمان إلى الجدول secemp من أجل كل سجل من سجلات جدول emp

فعلى سبيل المثال

```
SQL>INSERT INTO secemp  
(empno, name, job, mgr, sal, comm, deptno)  
(Select (mpno, name, job, mgr, sal, comm, deptno)  
From emp;
```

2- حذف بيانات موجودة في قاعدة البيانات (Deleting Record)

نستخدم عبارة DELETE لحذف سجلات من جدول ما ، وتأخذ الصيغة العامة لعبارة DELETE الشكل التالي :

```
SQL>DELETE [FROM] table  
[WHERE condition];
```

مما سبق نجد أن الصيغة العامة لعبارة Delete تتألف مما يلي :
1. .Sql: هي أحد الكلمات المحفوظة في Delete

2. From : هي أحد الكلمات المحفوظة في Sql ويليها إسم الجدول المراد حذف السجلات

منه

3. WHERE: هي أحد الكلمات المحفوظة في Sql ويليها شرط حذف السجلات ، وهي عبارة اختيارية للحد من عدد السجلات المحفوظة ، فإذا تم حذف عبارة where من جملة delete فإن هذا يعني حذف كافة السجلات

1-2 حذف سجلات محددة (Deleting SEPECIFIC Record)

مثال 1

لـحـذـف السـجـلـات الموـظـفـين ذـوي المرـتـبـات أـكـبر من 5000 من جـدول emp يـتم كـتابـة

الـعـبـارـة التـالـيـة

```
SQL> DELETE FROM emp  
      Where sal > 5000;
```

2-2 حـذـف كـافـة السـجـلـات (Deleting ALL Record)

يمـكـنـنـا حـذـف كـافـة السـجـلـات من جـدول ما بـحـذـف عـبـارـة where مـن جـملـة delete فـإن هـذـا

كـافـة السـجـلـات ، والـصـيـغـة العـامـة لـهـا هـيـ:

مثال

```
SQL>DELETE [FROM] table;
```

لـحـذـف كـافـة السـجـلـات من جـدول emp يـتم كـتابـة العـبـارـة التـالـيـة

```
SQL> DELETE FROM emp;
```

2-3 إـسـتـخـدـام الـاسـتـعـلـام الفـرـعـي في جـملـة DELETE

(**sub queries in DELETE statements**)

يمـكـنـنـا استـخـدـام الـاسـتـعـلـام الفـرـعـي في جـملـة Delete، وـذـلـك لـحـذـف سـجـلـات من جـدول

بنـاءـ عـلـيـ قـيـمـ من جـدول آـخـرـ، فـعلـيـ سـبـيلـ المـثـالـ إـذـا أـرـدـنـا حـذـف سـجـلـات الموـظـفـين التـابـعـين لإـدـارـةـ

المـبـيـعـاتـ يـتمـ ذـلـكـ عـلـيـ النـحوـ التـالـيـ

مثال

```
SQL> DELETE FROM employee
  2 WHERE      deptno =
  3          (SELECT      deptno
  4              FROM      dept
  5              WHERE      dname='sales');
5 rows deleted.
```

باستخدام Delete نستطيع حذف بيانات جدول واحد فقط في نفس الوقت.

- ✓ تزحف Delete جميع السجلات التي تحقق الشرط في جملة .Where
- ✓ لا تزحف Delete الجدول.
- ✓ تزحف السجلات من الجدول بشكل مستمر ، إلا أن الحذف الحقيقي للسجلات يعتمد على معالجة commit المستخدمة التي سيتم تناولها في هذا الفصل
- ✓ قد تسبب delete مشكلة في التجانس والتكامل المرجعي (وسيتم تناول ذلك عند إنشاء العلاقات بين الجداول لاحقاً في هذا الفصل).
- ✓ تزحف delete السجل أو السجلات بالكامل فلا حاجة للإشارة إلى أي عمود إلا في جملة Where عند استخدامها.

2- تعديل البيانات الموجودة في قاعدة البيانات

(Using the update command)

يمكننا استخدام update لتعديل القيم الموجودة في جدول ما ، والصيغة العامة لعبارة update تأخذ الشكل التالي:

```
SQL> UPDATE      table
    SET column = value [, column = value]
    [WHERE          condition];
```

ما سبق نجد أن الصيغة العامة لعبارة update تتألف مما يلي:

1. من الكلمات الممحوزة في، وهو أمر بتحديث البيانات في SQL.

2. Table: ويشير إلى إسم الجدول المراد التعديل في بياناته.

3. Set : وتحتوي على سلسلة من عمليات التحديث ، حيث نقوم بوضع إسم العمود

الذي نقوم بتعديل بياناته كمعامل أول ، ويحتوي المعامل الثاني على القيمة الجديدة ،

ويفصل بين المعاملين إشارة المساواة .

4. Where : وهي عبارة اختيارية يتم استخدامها لتحديد السجلات التي يراد التعديل بها ،

إذا تم حذف عبارة Where فإن التعديل سيتم على كل سجلات الجدول.

(Update ALL Record)

يمكننا تحديث كافة السجلات من جدول ما بحذف عبارة where من جملة Update فإن هذا

يعني تحديث كافة السجلات، والصيغة العامة لها هي:

```
SQL> UPDATE      table
    SET      column = value [, column = value];
```



مثال

لتحديث كافة السجلات من جدول employee وذلك بوضع المرتب 3000 لكل الموظفين يتم كتابة العبارة التالية:

```
SQL> UPDATE employee  
  2 SET          sal = 3000;  
14 rows updated.
```

3-2 تحديث سجلات محددة (Update Specific Record)

يمكننا تحدث سجلات محددة والتعديل فيها بناء على شرط أو عدة شروط، وذلك بتضمين عبارة Update فقرة where، فعلى سبيل المثال لتعديل مرتب الموظف رقم 555 من 3000 إلى 6000 يتم ذلك على النحو التالي:

```
SQL> UPDATE emp  
  2 SET          sal = 6000  
  3 WHERE         empno = 555;  
1 row updated.
```

3-3 استخدام الاستعلام الفرعى في حملة update

(sub queries in update statements)

يمكننا استخدام الاستعلام الفرعى في حملة update على سبيل المثال إذا أردنا تعديل أو تحدث كلا من الوظيفة والإدارة للموظف رقم 555 لتماثل الوظيفة والإدارة للموظف رقم 7489، سوف نحتاج إلى استخدام استعلام فرعى، ويتم كتابة عبارة update على النحو التالي:

```

SQL> UPDATE emp
  2 SET      (job, deptno) =
  3          (SELECT job, deptno
  4             FROM   emp
  5            WHERE  empno = 7489)
  6 WHERE    empno = 555;
1 row updated.

```

٤- استخدام تعليمات التثبيت commit والتراجع Rollback

يتطلب تفريغ التغييرات في بيانات الجداول خطوتين:

.Insert, Update, Delete) ١ تفريغ إحدى العبارات التالية

.commit) ٢ تأكيد التغيير باستخدام

عندما يتم تفريغ عبارة من عبارات DML، يقوم أوراكل بتغيير الجدول وحفظ نسخة من السجلات قبل إجراء التعديل، وسيبدو الأمر للمستخدم وكأن التغييرات قد تمت بشكل نهائي دائم، فإذا قام المستخدم بالاستعلام في قاعدة البيانات، فإنه سيرى التعديلات موجودة، فإذا قام مستخدم آخر بالاستعلام في قاعدة البيانات فلن يرى تلك التعديلات، وذلك لأن التعديلات على البيانات لا تصبح مرئية من قبل المستخدمين الذين يسمح لهم بالوصول إلى تلك الجداول، ولا تصبح دائمة حتى يتم تفريغ تعليمة COMMIT، بمعنى أن تلك الخاصية تسمح للمستخدم بالتراجع عن أي تغيير قام به. ويتم التراجع عن التعديلات باستخدام تعليمة Rollback. حيث تعيد هذه التعليمة قاعدة البيانات إلى الحالة التي كانت عليها بعد تفريغ آخر تعليمة commit.

تلميح

- ✓ يتم تنفيذ تعليمة `commit` بشكل ضمني إذا قطع المطور الاتصال مع نظام أوراكل بشكل نظامي أو نفذ تعليمة من تعليمات DDL، حيث يؤدي ذلك إلى حفظ التغييرات.
- ✓ إذا قطع الاتصال مع نظام أوراكل بشكل غير نظامي قبل تنفيذ تعليمة `COMMIT` لن يتم حفظ التغييرات، وستعود قاعدة البيانات إلى النقطة التي تم عدها تنفيذ آخر تعليمة `COMMIT`.
- ✓ يمكن التراجع عن التغييرات بشكل جزئي، حيث تسمح تعليمية `SAVEPOINT` للمستخدم حفظ التغييرات حتى جزء محدد من التغييرات، فعلى سبيل المثال نفترض أنك قمت بمجموعة من التعديلات على مجموعة من السجلات، ولكنك غير متأكد من أنك أجريت التعديلات بشكل صحيح، وترغب في رؤية السجلات قبل تثبيت هذه التغييرات، ففي هذه الحالة يمكنك استخدام عدة تعليمات `Savepoint`، وبالتالي يمكنك التراجع عن التغييرات حتى أي نقطة من النقاط العلام الموضوعة كما هو موضح بالشكل التالي ، والصيغة العامة لكلاً من نقطة العلام `Savepoint` والتراجع `Rollback` إلى نقطة علام تأخذ الشكل التالي :

```
SAVEPOINT save_point_name
ROLLBACK save_point_name
```

مثال 1

لتحديث كافة السجلات من جدول emp وذلك بوضع المرتب 3000 لكل الموظفين يتم كتابة العبارة التالية:

```
SQL> UPDATE employee  
  2 SET          sal = 3000;  
14 rows updated.
```

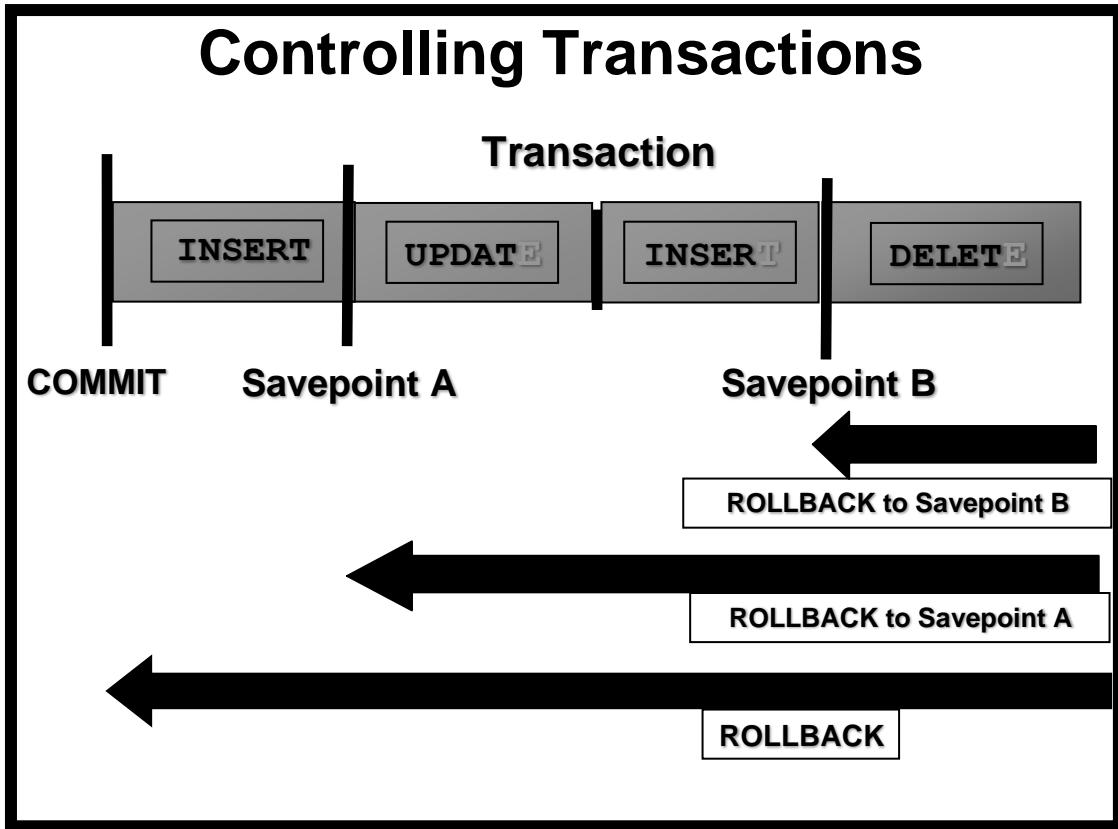
ولحفظ التعديلات في قاعدة البيانات يتم كتابة العبارة التالية:

```
SQL> COMMIT;  
Commit complete.
```

حذف كافة السجلات من جدول emp ثم التراجع عن ذلك يتم بكتابة العبارات التالية:

مثال 2

```
SQL> DELETE FROM emp;  
14 rows deleted.  
SQL> ROLLBACK;  
Rollback complete.
```



مثال 3

```

SQL> UPDATE...
SQL> SAVEPOINT jicc_update;
Savepoint created.
SQL> INSERT...
SQL> ROLLBACK TO jicc_update;
Rollback complete.

```

5- كيفية إنشاء جداول وإضافة محددات من نوع

(primary key, foreign key)

مما سبق نجد أنه يوجد في لغة SQL ثلاثة عبارات تسمى بعبارات تعريف البيانات وهي

تسمح لنا بتعريف ومحى وتعديل الجداول في قاعدة البيانات، وهذه العبارات هي:

1. **CREATE**: تستخدم لتعريف جدول جديد في قاعدة البيانات.

2. **DROP**: تستخدم لمحى جدول موجود من قاعدة البيانات.

3. تستخدم لـ**تغيير بنية الجدول** الموجود.: ALTER

قواعد البيانات العلائقية قادرة على تفزيذ كافة عبارات SQL في أي وقت بشرط أن يكون

له الحق في تنفيذ تلك الأوامر والعبارات، فلكي تقوم بإنشاء ومحى وتعديل الجداول لابد

من أن تتأكد من مدير قاعدة البيانات أنه يمنحك حق تعريف ومحى وتعديل الجداول

5-1 إنشاء وتعديل وإسقاط الجداول

١-١-٥ إنشاء الجداول (create table)

تمكنا عبارة `Create` من إنشاء الجداول في SQL، وتأخذ عبارة `Create` الشكل التالي:

```
SQL> CREATE TABLE [schema.]Table  
        (column datatype [DEFAULT expr]);
```

ما سبق نجد أن الصيغة العامة لعبارة Create تتألف مما يلي:

create -1: من الكلمات المحفوظة في Sql، وتخبر DBMS بإنشاء كائن في قاعدة

البيانات

Table-2: من الكلمات المحفوظة في Sql، وتخبر DBMS بنوع الكائن المراد إنشائه

في قاعدة البيانات

Table name-3: ويستخدم لتحديد إسم الجدول ، ويجب أن يبدأ إسم الجدول

حرف نصي ويمكن أن يحتوي أحرف وأرقام والشريطة، ويجب ألا يتجاوز طوله 30

.(underscore) حرف ، ولا يسمح بالفagrات ، ولكن يسمح بالشرطـة السفلـية

Column names-4: وتستخدم لتحديد أسماء الأعمدة المكونة للجدول المراد

إنشاءء، وتنطبق نفس القواعد الخاصة بالتسمية على إسم الجدول على أسماء الأعمدة

، ويجب أن تحاط مجموعة الأعمدة بين قوسين.

Column data type-5: يحتوي تعريف الأعمدة على نوع البيانات ، وطولها

و蜓تها كما هي موضح بالجدول التالي

Constraints-6: وهي إعدادات اختيارية يمكن استخدامها لتمثيل قيود للمحافظة

على تكامل قاعدة البيانات.

وهناك عدة اعتبارات يجب مراعاتها عند إنشاء الجداول وهي كالتالي:

1- يجب وضع أقواس تضم أسماء الأعمدة ونوع بياناتها.

2- يتم وضع فاصلة (,) بين تعريف أي عمود وأخر .

3- يجب أن يكون إسم العمود فريدا داخل الجدول.

4- لايمكن استخدام الكلمات المحظوظة كأسماء أعمدة في الجدول.

5- يتم استخدام الفاصلة المنقطة لإخبار SQL أن الجملة انتهت وجاهزة للتنفيذ.

6- الجدول التالي يوضح لنا أهم أنواع البيانات التي يمكن أن نستخدمها عند إنشاء

الجدول في SQL

الوصف	نوع البيانات
يعرف هذا النوع من البيانات عمود من نوع البيانات الحرفية بطول (n) حرف، وجميع الحقول متساوية في الطول، $n=255$	CHAR (n)

يعرف هذا النوع من البيانات عمود من نوع البيانات الحرفية بطول (n) حرف، وتخالف الحقول في الطول وتنتجاوز 255	VARCHAR (n)
يعرف عمود من نوع بيانات من نوع رقم، مع حد أقصى لعدد الأرقام (n)، ويمكن أن تأخذ (n) كحد أقصى 105	NUMBER (n)
يعرف عمود من نوع البيانات number، ويحتوي عدد أرقام قدره (n)، والتي تتضمن القيمة (d) التي توضح عدد الأرقام العشرية بعد الفاصلة.	NUMBER (n, d)
يشبه نوع البيانات NUMBER، ولكنه لا يأخذ غير القيم الرقمية الصحيحة، ولا يقبل أي أرقام عشرية بعد الفاصلة.	INTEGER
يعرف عمود من نوع البيانات الحرفية، وأقصى طول له 65.535، ولا يمكن تعريف سوي عمود واحد من هذا النوع في الجدول، ولا يمكن استخدام هذا العمود ضمن عبارات Where في الاستعلامات الفرعية أو الدوال أو التعبير أو ضمن الفهارس.	LONG
يعرف هذا النوع عمود يحتوي سجل بيانات بطول قدره n	RAW (n)
يعرف عمود يخزن أرقاماً موجبة وسالبة صغيرة تقع في النطاق (-32, 767, +32, 767)	SIMALLINT
يعرف عمود نوع بياناته تاريخ.	DATE

مثال 1

نقوم بإنشاء جدول department، الموضح بالكود التالي أسماء أعمدته ونوع بياناتها

```
SQL> CREATE TABLE department
  2 (deptno NUMBER (2) not null,
  3      dname    VARCHAR2 (15),
  4      loc      VARCHAR2 (20));
Table created.
```

ولعرض مواصفات وتعريفات الأعمدة نستخدم عبارة `describe` أو اختصارا `desc` وتأخذ الشكل التالي:

```
SQL> describe Table name;
```

Name	Null	Type
DEPTNO	NOT NULL	NUMBER(2)
DNAME		VARCHAR2(15)
LOC		VARCHAR2(20)

1-1-1-1 إنشاء الجداول باستخدام الاستعلام الفرعي

يمكنك إنشاء جدول باستخدام الاستعلام الفرعي كما بالمثال التالي:

```
SQL> CREATE TABLE dept10
  2 AS
  3     SELECT      empno, ename, sal*12 ANNSAL, hiredate
  4     FROM        emp
  5     WHERE       deptno = 10;
Table created.
```

نلاحظ وجود كلمة `as` بين إسم الجدول الجديد المراد إنشائه والاستعلام الفرعى، للتأكد من مواصفات وتعريفات الأعمدة نستخدم الأمر `describe column definitions`

SQL> `describe dept10;`

Name	Null	Type
EMPNO		NUMBER(4)
ENAME		VARCHAR2(10)
ANNSAL		NUMBER
HIREDATE		DATE

5-1-2 تعديل الجدول (altering the table)

يستخدم `alter table` في تعديل بنية جدول موجود وتغيير محددات الجدول بعد إنشائه

وفيها يلي الخيارات التي يمكن استخدامها مع الأمر `:alter`

`Add` : يستخدم هذا الخيار لإضافة أعمدة جديدة أو شروط إلى الجدول ويأخذ الشكل (1)

التالي:

```
SQL> ALTER TABLE table
ADD (column datatype [DEFAULT expr]
      [, column datatype] ...);
```

`Modify` (2) : ويستخدم لتغيير محددات موجودة ويأخذ الشكل التالي:

```
SQL>ALTER TABLE table
  MODIFY (column datatype [DEFAULT expr]
          [, column datatype] ...);
```

Disable (3): يستخدم لإلغاء تفعيل شرط ما في الجدول.

Enable (4): يستخدم لإعادة تفعيل شرط كنا قد قمنا بإلغاء تفعيله.

Drop (5): لإزالة شرط ما بشكل دائم من الجدول.

Set unused (6): يستخدم لتحديد عمود أو أكثر كأعمدة غير مستخدمة ، بحيث

يمكن حذف تلك الأعمدة من الجدول عندما تصبح مصادر النظام ملائمة لذلك.

Drop column (7): يستخدم لإزالة عمود ما من الجدول

مثال 1

لإضافة عمود job إلى الجدول dept10 نقوم بكتابة الكود التالي:

```
SQL> ALTER TABLE dept10
  2 ADD          (job VARCHAR2(9));
Table altered.
```

مثال 2

لتتعديل في نوع البيانات data type الـ ename في جدول dept10 في العمود

نقوم بكتابة الكود التالي:

```
SQL>ALTER TABLE dept30
  MODIFY      (ename VARCHAR2(15));
Table altered.
```

3-1-5 حذف الجداول (Dropping table)

تستخدم SQL عبارة drop لحذف الجداول من النظام، فكل ما تحتاجه لحذف جدول هو أن تكتب اسمه بعد drop table فيتم حذف الجدول نهائياً بجميع سجلاته وفهارسه، ولا يمحنا النظام فرصة لاستعادة الجدول، وتأخذ جملة Drop الشكل التالي:

SQL> DROP TABLE table_name;

مثال

لحذف جدول dept10 نقوم بكتابة الكود التالي

SQL> DROP TABLE dept10;

5-2 إضافة محددات إلى الجدول من نوع

(Primary key, foreign key)

5-2-1 المحافظة على تكامل قاعدة البيانات

يوجد في أوراكل بعض الأدوات لحماية قاعدة البيانات، تسمى هذه الأدوات بالشروط

، وتقوم هذه الشروط بعدة وظائف هامة منها:

1) التأكد من المفتاح الأساسي أو الرئيسي .unique

2) فرض التكامل المرجعي بمعنى التأكد من أن السجلات الأبناء الموجودة في جداول

مرتبطة، تمتلك سجل أب.

3) التأكد من أن الأعمدة المرتبطة بالشروط تحتوي دائماً قيمة بداخلها NOT NULL .

4) التأكد من وضع القيمة الافتراضية DEFAULT VALUE في عمود ما

5) التأكد من عموداً ما يحتوي على قيمة ما، وأن هذه القيمة موجودة ضمن نطاق محدد

من القيم The Check Constraint

1-1-2-5 الشرط CHECK (فحص وجود قيمة)

يستخدم هذا الشرط للتأكد من أن قيمة ما لعمود ما تقع بين قيم مجموعة محددة، عندما تستخدم الشرط CHECK سيقوم أوراكل بمقارنة أية قيمة يتم إدخالها مع مجموعة القيم المحددة في المجموعة.

مثال

نستخدم الشرط CHECK في تعريف جدول employee التالي ،تم إعطاء إسم للشرط التسمية الشرط يكون اختياريا، فإذا لم يتم تسمية الشرط فإن أوراكل سيولد إسماً خاصاً للشرط.

```
SQL> CREATE TABLE employee
  2  (emp_id NUMBER (2) not null,
  3           name      VARCHAR2 (15),
  4           gender    char (1)
  5   Constraint gender_validation check (gender
  6   in ('M', 'F')));
```

نلاحظ أن أوراكل سيقوم بمقارنة أية قيمة يتم إدخالها مع مجموعة القيم المحددة في المجموعة ('F', 'M') فقط في العمود gender بمعنى لا يسمح إلا بهاتين القيمتين فقط.

1-1-2-5 خيار القيمة الافتراضية (default option)

يستخدم هذا الخيار `default` لإعطاء قيمة افتراضية لعمودما عندما لا يتم وضع قيمة في هذا العمود، ولا يمنع هذا الخيار من جعل القيمة `null` في هذا العمود



مثال

في هذا المثال تم استخدام الخيار `default` للتأكد من العمود `comm` تم إعطائه القيمة صفر عند عدم وضع أي قيمة به، ويكون ذلك مفيدة في معالجة القيم الفارغة `null` عند إجراء العمليات الحسابية.

```
SQL> CREATE TABLE employee
  2  (emp_id NUMBER (2) not null,
  3      name      VARCHAR2 (15),
  4      gender    char (1)
  5      sal       number
  6      comm      default (0),
  Constraint gender_validation check (gender
in ('M', 'F'));
```

5-2-1-3-الشرط not null (ليس فارغاً)

نستخدم الشرط `not null` للتأكد من أن العمود دوماً سيحتوي قيمة بداخله، وخاصة ما يتم وضع هذا الشرط على عمود أو أعمدة المفتاح الأساسي للجدول ، لكي تقوم بوضع الشرط `not null` على عمود ما ،قم بوضع العبارة `not null` بعد نوع بيانات العمود كما بالمثال التالي:

```
SQL> CREATE TABLE employee
  2  (emp_id NUMBER (2) not null,
  3      name      VARCHAR2 (15),
  4      gender    char (1)
  5      sal       number
  6      comm      default (0),
  Constraint gender_validation check (gender
in ('M', 'F'));
```

4-1-2-5 الشرط unique (وحيد)

يستخدم الشرط `unique` للتأكد من أن القيمة الموجودة بالعمود المشروط هي قيمة وحيدة في جميع سجلات الجدول، بمعنى عدم السماح بتكرار القيم، فيقوم هذا الشرط بعمله عبر إنشاء فهرس وحيد (`unique index`) على هذا العمود، يعتبر الشرط `unique` أداة جيدة للمحافظة على هذه الخاصية في قاعدة البيانات، فعلى سبيل المثل يمكن اعتبار عمود `number_social_security`" الضمان الاجتماعي " الموجود في جدول `employee` لا يحتوي سوى قيمة وحيدة، مع العلم بأن العمود " `emp_id` " هو المفتاح الأساسي في الجدول بمعنى أنه لابد أن يحتوي قيماً فريدة ووحيدة ولا يسمح بالتكرار كما هو موضح كالتالي:

```
SQL> CREATE TABLE employee
  2  (emp_id NUMBER (2) not null
  3    constraint unique_emp_id unique,
  4      name      VARCHAR2 (15),
  5      gender    char (1)
Constraint gender_validation check (gender
```

5-1-2-5 الشرط primary key (المفتاح الأساسي)

يستخدم الشرط `primary key` للمحافظة على تكامل عمود أو أعمدة المفتاح الأولي، حيث يجعل هذا الشرط القيم الموجودة في المفتاح في عمود مشروط به تتمتع بالشروطين

unique معًا، فعند تعريف هذا الشرط يتم إنشاء فهرس وحيد unique .
 ضمني وإنشاء شرط not null ضمني أيضًا على العمود أو الأعمدة المشروطة index .
 بشرط المفتاح الأساسي، ويمكن اعتبار تعريف هذا الشرط كجزء من تعريف العمود أو جزء من تعريف الجدول ، حيث إذا تم تعريف هذا الشرط على عدة أعمدة (حالة مفتاح أساسي مركب) فإن تعريف هذا الشرط سيكون جزءاً من تعريف الجدول.

مثال

لكي تعرف الشرط primary key كجزء من تعريف عمود ما، قم بكتابة العبارة primary key بعد نوع بيانات الجدول كما بالكود التالي:

```
SQL> CREATE TABLE employee
  (emp_id NUMBER primary key,
   name   VARCHAR2 (15),
   Gender  char (1)
  Constraint gender_validation check (gender
  in ('M','F')));
```

مثال

لكي تعرف الشرط primary key كجزء من تعريف الجدول، قم بكتابة العبارة primary key في أسفل تعريف الجدول، ويتم تعريفه بعد تعريف آخر عمود، ويجب وضع فاصلة بعد تعريف آخر عمود وقبل تعريف شرط المفتاح الأساسي كما بالكود التالي:

```
SQL> CREATE TABLE employee
  (emp_id NUMBER not null,
   F_name      VARCHAR2 (25),
   L_name      VARCHAR2 (15),
   Primary key(emp_id));
```

6-1-2-5 الشرط foreign key (المفتاح الخارجي)

يستخدم الشرط foreign key كأداة للتأكد من الجداول المرتبطة لا تحتوي على سجلات غير منتمية إلى سجل أب، ويحتوي الشرط foreign key على خيار يسمح بحذف كل السجلات الأبناء المرتبطة مع سجل الأب عند حذف سجل الأب وهذا الخيار هو ON DELETE CASCADE، يمكننا إنشاء مفتاح الرابط الخارجي كجزء من تعريف العمود عند إنشاء الجدول ، حيث يتم استخدام الكلمة REFERENCES ويليها إسم الجدول الذي يحتوي السجل الأب وذلك يكون بعد تعريف العمود ولسنا بحاجة لذكر إسم العمود من الجدول الأب ، حيث إذا لم تقوم بكتابه إسم العمود الموافق من الجدول الأب يفترض أوراكل أنه هو المفتاح الأساسي

مثال 1

في هذا المثال سنقوم بتعريف الشرط Foreign key كجزء من تعريف عمود ما كما هو موضح التالي:

```
SQL> CREATE TABLE consultant_projects
  2  (emp_id number references consultant,
  3  project_name      varchar (25),
  4  complete_date     date);
Table created
```

مثال 2

لكي تعرف الشرط `foreign key` كجزء من تعريف الجدول، قم بكتابة العبارة `foreign key` في أسفل تعريف الجدول متبوعة باسم العمود المطبق عليه الشرط ثم كلمة `reference` ثم عمود المفتاح الرئيسي بالجدول الأب، ويتم تعريفه بعد تعريف آخر عمود، ويجب وضع فاصلة بعد تعريف آخر عمود وقبل تعريف شرط المفتاح الخارجي كما بالكود التالي:

```
SQL> CREATE TABLE emp (
 2      empno      NUMBER(4),
 3      ename      VARCHAR2(10) NOT NULL,
 4      job        VARCHAR2(9),
 5      mgr        NUMBER(4),
 6      hiredate   DATE,
 7      sal        NUMBER(7,2),
 8      comm       NUMBER(7,2),
 9      deptno    NUMBER(7,2) NOT NULL,
10      CONSTRAINT emp_deptno_fk FOREIGN KEY(deptno)
11      REFERENCES dept (deptno);
```

7-1-2-5 تعديل تعريف شرط ما (Modifying Constraints)

يمكننا إضافة شرط للجدول وذلك بعد إنشائه، كما يمكننا إزالة تلك الشروط أو إلغاء تفعيلها أو تفعيلها وذلك باستخدام `alter table` ويملك هذا الأمر عدة خيارات يمكن استخدامها مع

الشروط وهي موضحة كالتالي:

(إضافة): يستخدم لإضافة شرط جديد إلى الجدول.

(تعديل): يستخدم لإضافة شرط إلى عمود موجود.

3 (Drop) : يستخدم لحذف الشرط من الجدول.

4 (Disable) : يسمح بإدخال البيانات دون النظر في فيما إذا كانت تتحقق

الشرط أم لا ، ولكن مع بقاء الشرط كما هو في قاموس البيانات.

5 (Enable) : يقوم بالتحقق من أن جميع البيانات التي يتم إدخالها والموجودة

مسبيقاً تحقق الشرط.

مثال 1

```
SQL> alter table consultant_projects  
  2 add primary key (emp_id,project_name);
```

مثال 2

مثال 2

```
SQL> ALTER TABLE EMP  
  2 ADD CONSTRAINT emp_mgr_fk  
  3 FOREIGN KEY (mgr) REFERENCES emp (empno);  
Table altered.
```

مثال 3

```
SQL> ALTER TABLE emp  
  2 disable primary key;  
Table altered.
```

مثال 4

```
SQL> ALTER TABLE emp  
  2 enable primary key;  
Table altered.
```

مثال 4

مثال 5

```
SQL> ALTER TABLE      emp
  2 drop primary key;
Table altered.
```

الأسئلة

المجموعة الأولى:

ضع علامة (✓) أمام العبارة الصحيحة وعلامة (✗) أمام العبارة الخاطئة مع تصحيح الخطأ.

1. عند استخدام Insert into لائحة القيم لابد أن تأخذ نفس ترتيب الأعمدة في

الجدول المراد إضافة السجلات إليه.

2. للتأكد من تنفيذ عبارة insert وإضافة السجل إلى جدول emp يقوم بتنفيذ العبارة

التالية:

```
SQL>Select *      From emp;
```

3. عند استخدام عبارة insert يجب أن تتطابق الأعمدة والقيم المقابلة لها في نوع البيانات

.data types

4. عند استخدام عبارة insert لا يتم وضع الأحرف النصية والتاريخ بين علامتي

تصنيص مفردة ' '.

5. لحذف كافة السجلات من جدول emp يتم كتابة العبارة التالية

SQL> DELETE * FROM emp;

6. لا يمكننا استخدام الاستعلام الفرعى في عبارة delete.

7. تزحف Delete الجدول والبيانات التي بداخله.

8. تزحف delete السجل أو السجلات بالكامل فلا حاجة للإشارة إلى أي عمود إلا في

جملة Where عند استخدامها.

9. لتحديث كافة السجلات من جدول emp وذلك بوضع المرتب 3000 لكل الموظفين

يتم كتابة العبرة التالية:

Sql> Update table emp set sal =3000;

10. نستخدم commit لحفظ وتأكيد التغييرات، بينما rollback للتراجع عن

التغييرات.

المجموعة الثانية:

أسئلة لتدريب الطالب على أسئلة الاختيار المتعدد:

1. عند استخدام insert:

a) يجب أن تتطابق الأعمدة والقيم المقابلة لها في نوع البيانات .data types

b) لائحة القيم لابد أن تأخذ نفس ترتيب الأعمدة في الجدول المراد إضافة السجلات إليه.

c) لابد من وجود علاقة واحد إلى واحد بين الأعمدة ولائحة القيم، بمعنى أن توجد

قيمة واحدة من أجل كل عمود وبنفس الترتيب.

d) جميع ما سبق.

2. عند استخدام insert:

(a) قد نجهل قيمة أحد الحقول، وذلك بكتابة كلمة NULL

(b) قد تسبب مشكلة في التجانس والتكامل المرجعي INSERT

(c) لابد من وضع الأحرف النصية والتاريخ بين علامتي تصييص مفردة ' ' .

A&b&c (d)

A&c (e)

3. عند استخدام `:delete`

(a) يمكننا استخدام الاستعلام الفرعى في جملة Delete .

(b) تزحف Delete جميع السجلات التي تحقق الشرط في جملة Where .

(c) تزحف Delete الجدول

a&b (d)

4. عند استخدام `:update`

(a) يمكننا استخدام update لتعديل القيم الموجودة في جدول ما.

(b) إذا تم حذف عبارة Where فإن التعديل سيتم على كل سجلات الجدول.

(c) لا يمكننا تحديث سجلات محددة والتعديل فيها بناء على شرط أو عدة شروط.

(d) لا يمكننا استخدام الاستعلام الفرعى في حملة update فعلى سبيل المثال إذا أردنا.

A&b (e)

5. عند استخدام تعليمية `:commit`

a. تأكيد التغيير باستخدام commit .

b. لا يتم تنفيذ تعليمية commit إذا قطع المطور الاتصال مع نظام أوراكل بشكل

نظامي

c. إذا قطع الاتصال مع نظام أوراكل بشكل غير نظامي قبل تنفيذ تعليمية

لأن يتم حفظ التغييرات COMMIT

d. يمكن التراجع عن التغييرات بشكل جزئي باستخدام SAVEPOINT حيث

تمكن المستخدم حفظ التغييرات حتى جزء

A&C&d .e

6. عند استخدام :-create

a) تمكنا عبارة SQL من إنشاء الجداول في

b) ليس من الضروري وضع أقواس تضم أسماء الأعمدة ونوع بياناتها.

c) يتم وضع فاصلة (,) بين تعريف أي عمود وأخر.

d) يجب أن يكون إسم العمود فريدا داخل الجدول.

e) يمكن استخدام الكلمات المحجوزة كأسماء أعمدة في الجدول.

A&C&d (f)

7. عند استخدام :-create

a) يمكنك إنشاء جدول باستخدام الاستعلام الفرعى

b) وضع كلمة as بين إسم الجدول الجديد المراد إنشائه والاستعلام الفرعى

c) يجب وضع أقواس تضم أسماء الأعمدة ونوع بياناتها

A&b&c (d)

8. من أهم الأسباب التي تدعونا إلى إنشاء :constraints

a) التأكد من المفتاح الأساسي أو الرئيسي .unique

b) فرض التكامل المرجعي بمعنى التأكيد من أن السجلات الأبناء الموجودة في جداول

مرتبطة، تمتلك سجل أب.

c) التأكيد من أن الأعمدة المرتبطة بالشروط تحتوي دائمًا قيمة بداخلها NOT NULL.

d) التأكيد من وضع القيمة الافتراضية DEFAULT VALUE في عمود ما.

e) جميع ما سبق.

9. عند استخدام الشرط :-**check**

a) يستخدم هذا الشرط للتأكد من أن قيمة ما لعمود ما تقع بين قيم مجموعة محددة

b) يقوم أوراكل بمقارنة أية قيمة يتم إدخالها مع مجموعة القيم المحددة في المجموعة

A&b (c

d) ليس شيء مما سبق.

10. الشرط :**not null**

a) نستخدم الشرط not null للتأكد من أن العمود دوماً سيحتوي قيمة بداخله

b) يتم وضع هذا الشرط على عمود أو أعمدة المفتاح الأساسي للجدول

A&b (c

d) ليس شيء مما سبق.

11. الشرط :**default**

a) يستخدم هذا الخيار default لإعطاء قيمة افتراضية لعمود ما

b) يمنع هذا الخيار من جعل القيمة null في هذا العمود

A&b (c

d) ليس شيء مما سبق.

12. الشرط :unique

(a) يستخدم الشرط unique للتأكد من أن القيمة الموجودة بالعمود المشروط هي قيمة

وحيدة في جميع سجلات الجدول.

(b) فيقوم هذا الشرط بعمله عبر إنشاء فهرس وحيد (unique index) على العمود

المشروط.

A&b (c

(d) ليس شيء مما سبق.

13. الشرط :primary key

(a) يستخدم الشرط primary key للمحافظة على تكامل عمود أو أعمدة المفتاح

الأولي.

(b) يجعل هذا الشرط القيم الموجودة في المفتاح في عمود مشروط به تتمتع بالشروطين

Not null, unique (c

(d) عند تعريف هذا الشرط يتم إنشاء فهرس وحيد.

(e) يمكن اعتبار تعريف هذا الشرط كجزء من تعريف العمود أو جزء من تعريف الجدول

(f) جميع ما سبق.

14. الشرط :foreign key

(a) يستخدم الشرط foreign key كأداة للتأكد من الجداول المرتبطة لا تحتوي على

سجلات غير منتمية إلى سجل أب.

(b) يحتوي الشرط foreign key على خيار يسمح بحذف كل السجلات الأبناء

المرتبطة مع سجل الأب

c) يمكننا إنشاء مفتاح الربط الخارجي كجزء من تعريف الجدول

d) جميع ما سبق

15. باستخدام :`alter table`

a) يمكننا إضافة شرط للجدول وذلك بعد إنشائه

b) كما يمكننا إزالة الشروط أو إلغاء تفعيلها أو تفعيلها

A&b (c)

d) ليس شيء مما سبق

الفصل الرابع

ورشة عمل

الأهداف:

إعطاء الطالب ترسیخ المفاهيم الأساسية لغة SQL في ذهن الطالب من خلال قيام الطالب بإنشاء عدة جداول، وإضافة محددات من نوع Primary key، foreign key، والقيام بعمليات إضافة البيانات إلى الجداول وحذف وتعديل البيانات في قاعدة البيانات، وإجراء الاستعلامات المختلفة على هذه الجداول

لمحتويات:

1) إعطاء الطالب تعريفات لعدة جداول ويطلب منه القيام بإنشائهما.

2) إضافة سجلات جديدة للجداول.

3) حذف بيانات موجودة في قاعدة البيانات.

4) تعديل البيانات الموجودة في قاعدة البيانات.

5) إضافة محدّدات من نوع (primary key, foreign key)

6) إنشاء العديد من الاستعلامات

ماذا سنتعلم في هذا الفصل:

في نهاية هذا الفصل يكون الطالب قد اكتسب المهارات والمعرفات التالية:

(1) إنشاء وتعديل وإسقاط الجداول

(2) إضافة سجلات جديدة للجداول.

(3) حذف بيانات موجودة في قاعدة البيانات .

(4) كيفية تعديل البيانات الموجودة في قاعدة البيانات.

(5) إضافة محددات من نوع (primary key, foreign

(6) التدريب على إنشاء الاستعلامات المختلفة.

ورشة عمل

1- قم بإنشاء الجداول التالية:

الجدول الأول: department

Name	Null	Type
DEPTNO	NOT NULL	NUMBER(2)
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)

الجدول الثاني: employee

Name	Null	Type
EMPNO	NOT NULL	NUMBER(4)
ENAME		VARCHAR2(10)
JOB		VARCHAR2(9)
MGR		NUMBER(4)
HIREDATE		DATE
SAL		NUMBER(7,2)
COMM		NUMBER(7,2)
DEPTNO		NUMBER(2)

2- قم بتنفيذ التعليمات التالية:

1) قم بإنشاء كلا الجدولين بناء على التعريفات الموضحة سابقاً.

2) في الجدول الأول department قم بإضافة الشرط Primary Key لحقل

empno في جدول employee، وإضافة الشرط deptno

.employee

3) إضافة الشرط Foreign key deptno لحقل deptno في جدول employee كجزء من

تعريف الجدول.

4) إنشاء الرابطة الذاتية في جدول employee حيث يشير عمود mgr إلى عمود المفتاح

الأساسي في الجدول وهو empno.

5) قم بإدخال البيانات التالية في جدول department

Dept no	dname	location
1	accounting	New York
2	sales	Boston
3	operations	Chicago

6) قم بإدخال البيانات التالية في جدول employee

empno	Ename	job	mgr	hiredate	sal	comm	deptno
1	ayman	accountant	4	199/8/5 9	300 0	500	1
2	Said	Sales man		200/9/5 1	250 0	250	2
2	Tamer	Sales man	2	199/5/2 9	400 0	530	2
4	ali	accountant		199/6/4 9	500 0		1
5	ahamed	accountant	4	199/8/2 8	350 0	120	1

7) قم باسترجاع بيانات الموظفين (empno, ename, sal) الذين يتلقاون مرتبًا

ينحصر بين 1000، 3000.

8) قم باسترجاع جميع بيانات الموظفين من جدول employee الذين تبدأ أسماؤهم بحرف

.a

(9) قم بإنشاء إستعلام لاسترجاع بيانات الاسم والوظيفة والمرتب للموظفين الذي تم تعينهم

خلال عام 1999 وذلك من شهر يناير إلى شهر ديسمبر.

(10) قم بإنشاء إستعلام لاستعادة بيانات جميع الموظفين مرتبة تنازليا على أساس تاريخ

التعيين.

(11) قم بتحديث بيانات المرتب للموظفين الذين يعملون بالإدارة رقم 2 وذلك بزيادة المرتب

بنسبة 5%.

(12) قم بكتابة العبارة التي تمكنا من حذف جميع سجلات جدول emp.

(13) قم بإضافة عمود address إلى جدول emp علمًا بأن نوع بياناته هي
.varchar(20)

(14) قم بإنشاء إستعلام يقوم بحذف سجلات الموظفين التابعين لإدارة المبيعات.

(15) قم بإضافة الشرط check للعمود address على ألا يقبل سوى سكان

.new york, cairo, alex, jeddah

الفصل الخامس

مصمم النماذج

Form Builder

الأهداف :

إعطاء الطالب صورة موضحة عن أساسيات منشئ النماذج Form Builder، وكيفية إنشاء النماذج باستخدام المعالج Form Wizard، والتعديل على الشكل العام للنموذج وإضافة الحقول إلى النموذج، مع تحديد وتعديل خصائص الكائنات على النموذج.

المحتويات:

oracle Developer (1) مقدمة عن

oracle Developer أدوات البناء في (2)

Form Builder استخدام منشئ النماذج (3)

المكونات الأساسية لمنشئ النماذج. (4)

إنشاء النماذج باستخدام المعالج. (5)

تنفيذ تطبيق منشئ النماذج (6)

7) خصائص الكائنات الموجودة على النموذج

ماذا سنتعلم في هذا الفصل:

في نهاية هذا الفصل يكون الطالب قد اكتسب المهارات والمعرفات التالية:

1) مقدمة عن oracle Developer

2) مكونات oracle Developer

3) استخدام منشئ النماذج form Builder

4) المكونات الأساسية لمنشئ النماذج.

5) إنشاء النماذج باستخدام المعالج.

6) تنفيذ تطبيق منشئ النماذج

7) التعرف على خصائص الكائنات الموجودة علي النموذج من النوافذ وكتلة البيانات والإطار

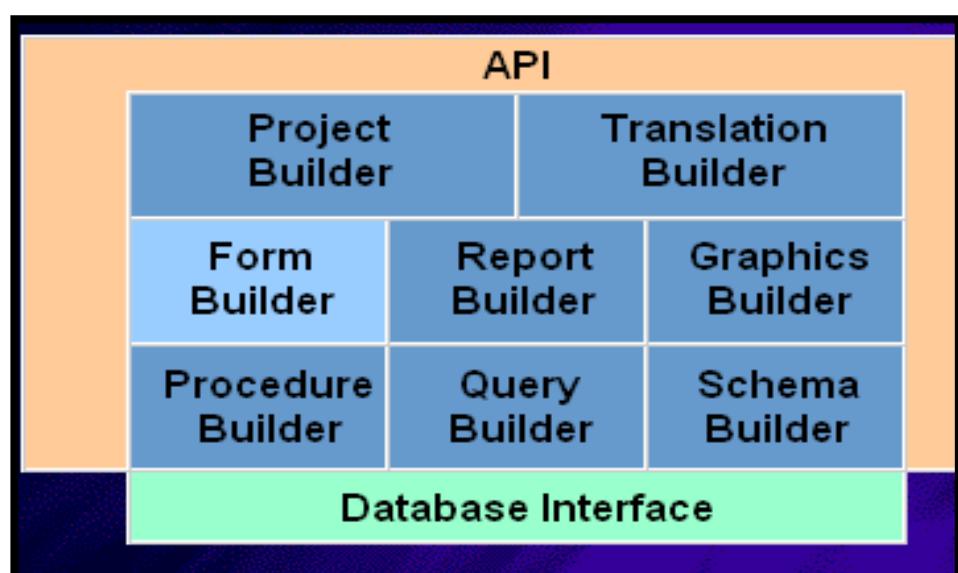
وعناصر النص.

oracle Developer a. مقدمة عن

ت تكون Oracle Developer من العديد من الأدوات القوية التي تسمح لك بإنشاء تطبيقات Client/Server تتميز بالقوة والدقة، وعالية التدرج بشكل أسرع بكثير من استخدام لغات البرمجة مثل C أو Visual Basic أو Java ويوفر لك أوراكل العديد من الأدوات تسمى بالمنشآت Builders لإنشاء الأجزاء المختلفة للتطبيقات كالنماذج Forms والتقارير Report والاسئلalas Queries والمخططات البيانية charts والإجراءات Procedures، وعناصر قواعد البيانات Database Objects، تزودنا oracle ب العديد من الطرق التي تمكنا من تحسين وعمل البرامج المختلفة ويأخذ المطور developer الشكل التالي :



ويحتوي oracle Developer على عدد من الأقسام التي تساعد علي إنشاء وتطوير والبرامـج في جميع المجالـات وتسهل لك أداء الوظائف البرمجية من خـلال بيـئة رسومـية GUI سهـلة الاستخدام والبرمـجة كما بالشكل التالي:



1-1 منشئ المشاريع project builder

يعتبر منشئ المشاريع نقطة بداية لأغلب أدوات oracle developer حيث

يتضمن منشئ المشاريع شريط أدوات يحتوي اختصارات لكلا من:

(1) Run Form Builder (منشئ النماذج)

(2) Run report Builder (منشئ التقارير)

(3) Run graphics Builder (منشئ الرسوم البيانية)

(4) Run procedure Builder (منشئ الوحدات البرمجية)

(5) Run Query Builder (منشئ الاستعلامات)

كما يمكنك من إنشاء ملفات مشاريع جديدة ، وفتح مشاريع موجودة مسبقاً

2-1 منشئ النماذج Form builder

ويساعدك على إنشاء شاشات إدخال البيانات والمعروفة بالنماذج، والنماذج عبارة عن

تطبيقات تقوم بالاتصال مع قاعدة البيانات واسترجاع البيانات المطلوبة من قبل المستخدم، وإظهارها

ضمن تخطيط تم تصميمه من قبل المصمم والسماح للمستخدم بإضافة وتعديل البيانات 3-1

منشئ التقارير Report Builder

وهو عبارة عن أداة تستخدم لإنشاء التقارير ضمن تطبيقات oracle ويساعدك في إنشاء

التقارير في مشروعك أو تطبيقك وربطها بالشاشات المختلفة من خلال بيئة رسومية سهلة الاستخدام

GUI ، وتساعد هذه التقارير علي استرجاع بيانات من قاعدة البيانات وفق تخطيط محدد من قبل

مصمم التقارير، غالباً ما تحتوي علي مجاميع جزئية Subtotals وتلخيصات

Summaries، غالباً ما تشمل علي رسوم بيانية لإعطاء مستخدم التقرير صورة أوضح عن بيانات هذا التقرير.

4- منشء الاستعلامات Query Builder

ويساعد علي إنشاء الاستعلامات المختلفة من خلال بيئة رسومية GUI، عوضاً عن استخدام معظم الأوامر النصية التي كنا نقوم بكتابتها في SQL، مما يسهل عملية البرمجة ويقلل الأخطاء.

5- منشء الرسوم البيانية Graphics Builder

ويساعد علي إنشاء الرسوم البيانية المختلفة من خلال بيئة رسومية GUI، وربطها بكلٍّ من النماذج والتقارير المختلفة.

6- منشء المخططات schema builder

وهو عبارة عن أداة من أدوات المطور 2000 تسمح لنا بالعديد من الوظائف من خلال بيئة رسومية سهلة منها تعريف الجداول Tables، المشاهد Views واللقطات Snapshot، وعلاقات الربط Relationships بين الجداول المختلفة في قاعدة البيانات، ومن ثم تغيير هذا التصميم، كما يسمح لنا منشء المخططات برؤية تصميم قاعدة البيانات .

6- منشء الإجراءات procedure builder

ويمكنك من إنشاء الوحدات البرمجية المختلفة والتي سنتعرض لها لاحقاً، وذلك من خلال بيئة رسومية تسهل لك إنشاء الوحدات البرمجية بسهولة.

7- منشء الترجمات Translation builder

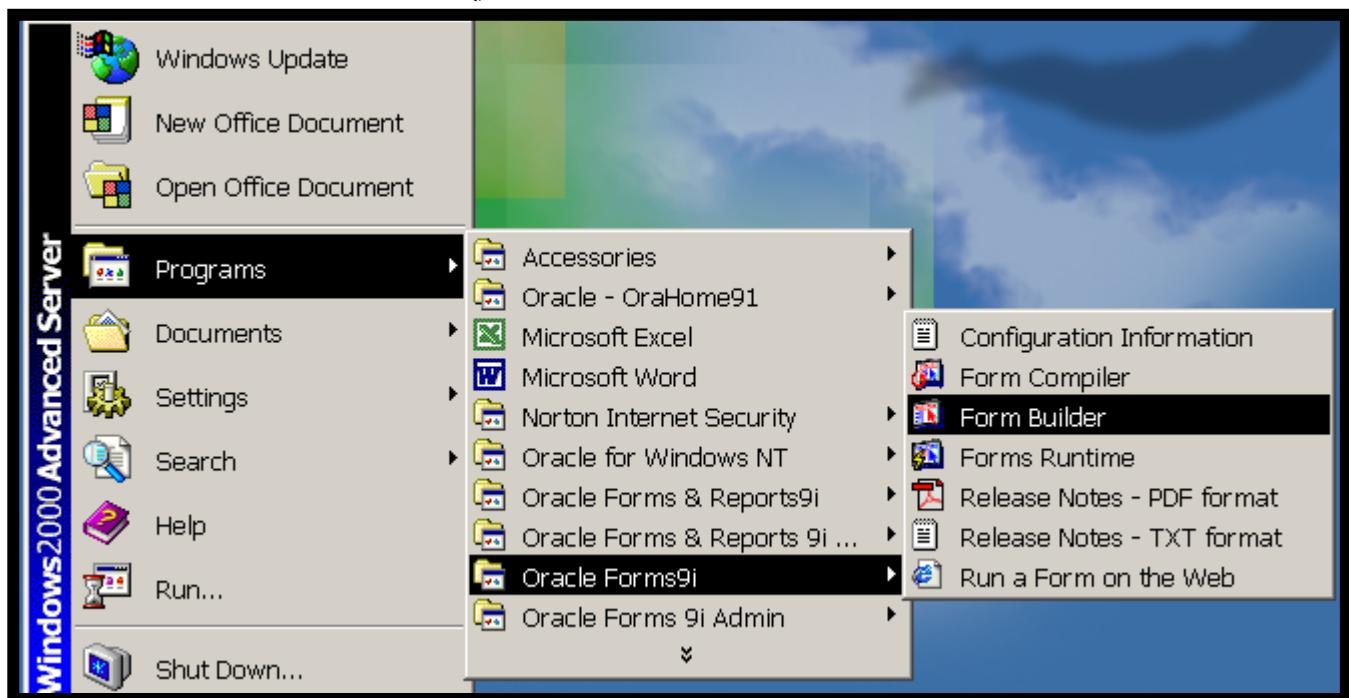
يساعد المبرمجين على ترجمة نصوص تطبيق ما إلى لغات برمجة أخرى، وحفظ هذه الترجمات في مخزن Repository يمكن استخدامه تلقائياً عند ترجمة الإصدارات القادمة للتطبيق.

2- منشئ النماذج Form Builder

هذا الجزء سيكون مجال دراستنا في هذا الفصل، وسنتناول المكونات الأساسية لمنشئ النماذج و إضافة العناصر الأساسية لوحدة النموذج، وإنشاء كتل البيانات Data Block، وإنشاء علاقه الربط RELATIONSHIP بين كتل البيانات، وتحديد وتعديل خصائص كل كائن موجود في النموذج.

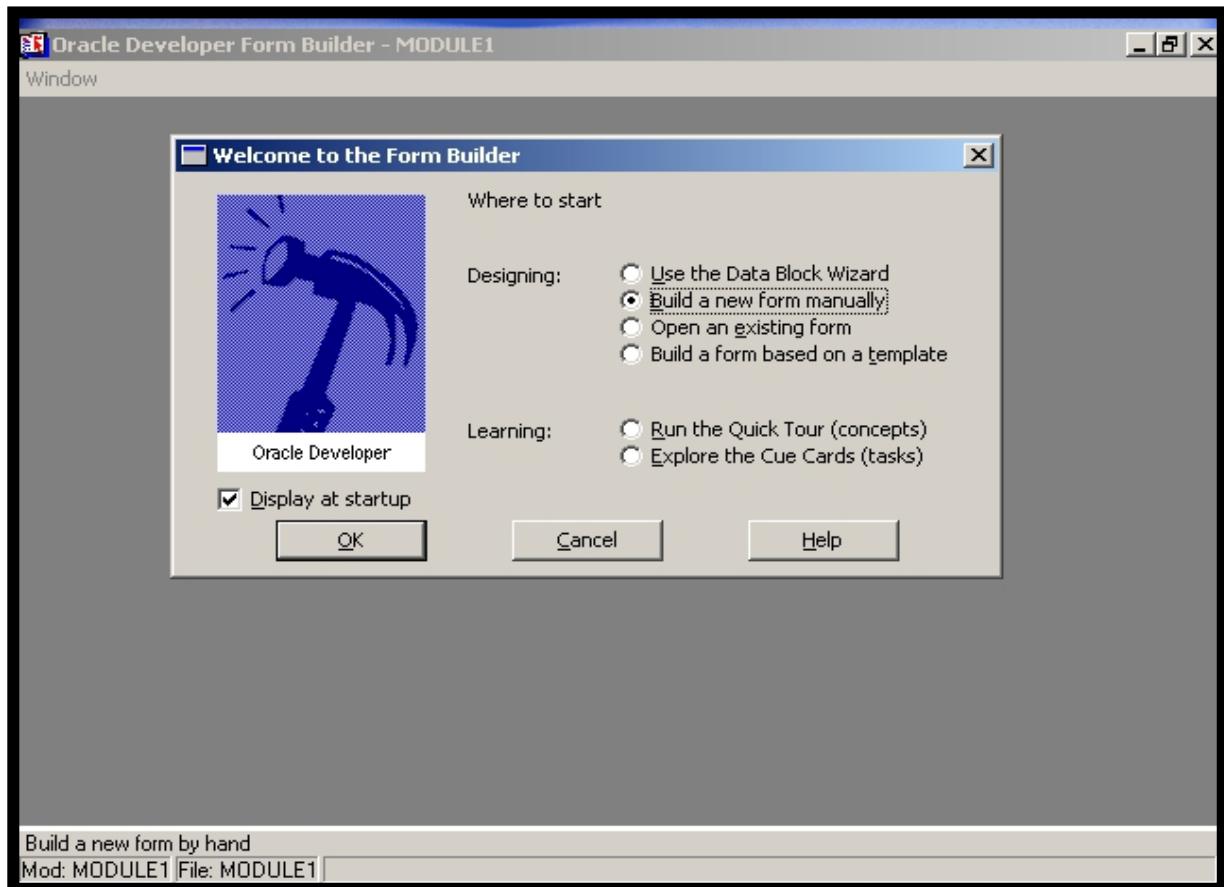
2-1 كيفية تشغيل منشئ النماذج Form Builder

من قائمة أبدأ يتم الإشارة إلى مجموعة البرامج ثم اختيار Oracle forms9i ثم اختيار منشئ النماذج Form builder كما هو موضح بالشكل التالي:



2-2 إنشاء وحدة نموذج Form Builder

وبالضغط على منشئ النماذج Form builder من قائمة البرامج يظهر لنا صندوق الترحيب التالي والذي من خلاله نستطيع تحديد الطريقة التي سيتم تصميم النموذج بها، كما هو موضح بالشكل التالي:



ونجد من الشكل السابق أن هناك عدة اختيارات عند تصميم النماذج، منها أربعة اختيارات في تبويب Designing، و اختياران لمساعدة في تبويب Learning، وهي موضحة كالتالي:

Use the data block wizard (1)

ويوفر هذا الاختيار معالج يساعدنا في إنشاء النماذج، ويتم اختياره عندما نريد إنشاء

نموذج آلياً باستخدام معالج من قبل البرنامج يسهل لنا عملية الإنشاء

:Build anew Form Manually (2)

يمكنا عن طريق هذا الاختيار من إنشاء النموذج بطريقة العرض التصميمي، بمعنى أن المبرمج سيقوم بإنشاء النموذج يدوياً.

: Open an existing form (3

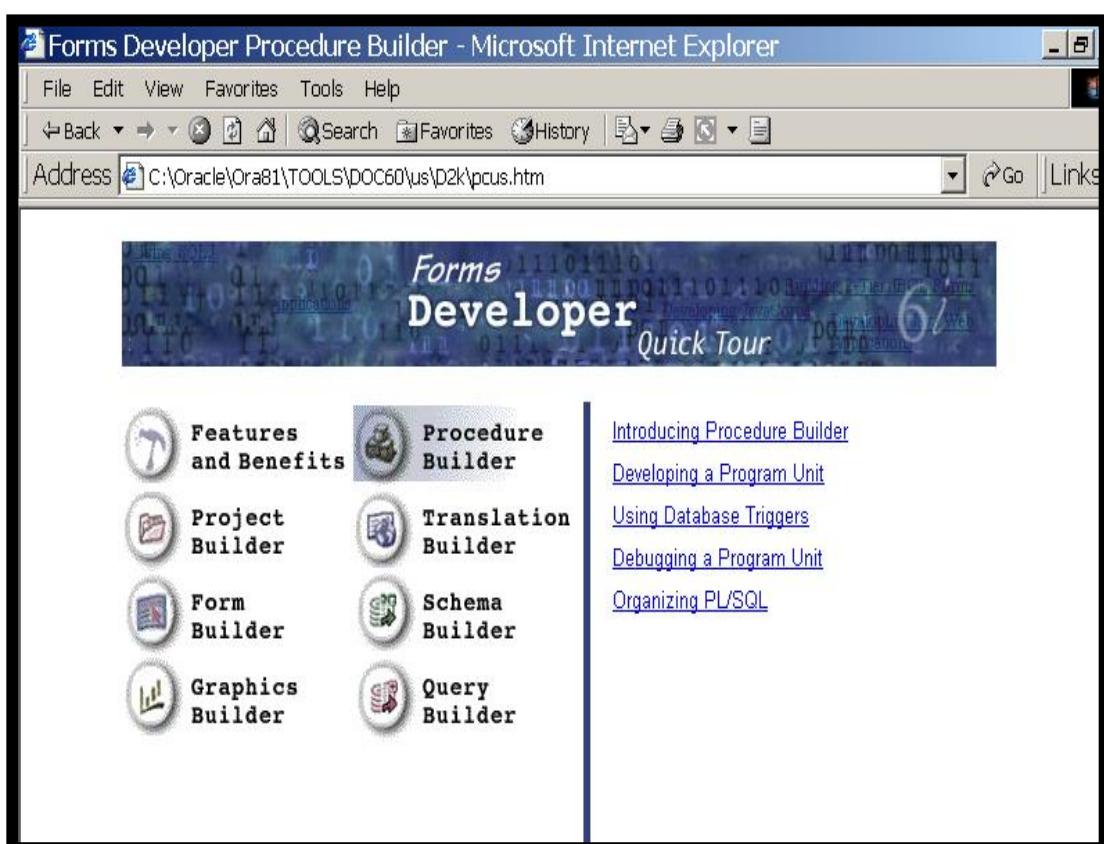
ويمكننا هذا الاختيار من فتح نموذج موجود من قبل.

:Build a form based on template (4

يمكننا هذا الاختيار من إنشاء نموذج بناء على قالب نموذج آخر.

(5) الاختيارات في تبويب Learning عن طريقهما يتم الوصول إلى مساعد المطور

كما هو موضح بالشكل التالي:

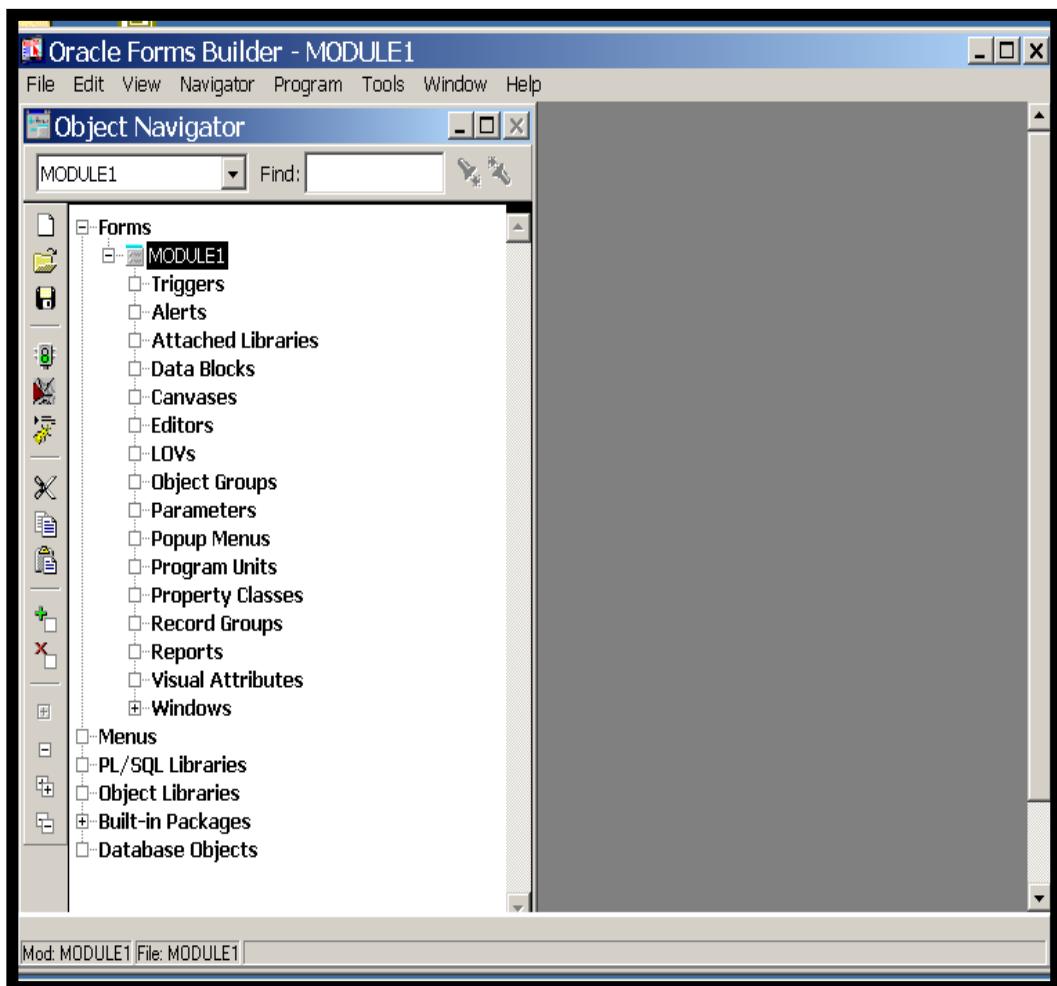


2-3 المكونات الرئيسية لمنشئ النماذج

(Components of form builder)

يحتوي منشئ النماذج form builder على العديد من المكونات المختلفة التي يزودنا

كلاً منها بعمل محدد، وتسمى هذه المكونات modules كما بالشكل التالي:



ويمكن تجميع هذه المكونات في عدة فئات كما بالشكل السابق:

(a) وحدات النموذج form modules: وتكون أغلب جسم التطبيقات، وعن طريقها

يمكننا تصميم شاشات إدخال البيانات وشاشات الاستعلامات المختلفة والشاشات

الرئيسية للتطبيقات المختلفة.

(b) وحدات القائمة: `Menu Modules` وتتضمن تخزين المعلومات عن القوائم

المخصصة التي قمت بإنشائها لتطبيقك، بالإضافة إلى تخزين التعريف والاكواد

المختلفة بعناصر القوائم، وتحفظ وحدات القائمة بمسار المكتبات `Libraries`

التي تستخدمها القائمة وغيرها وسوف تتعرض لذلك لاحقا.

(c) وحدات مكتبة `pl/sql` وتحتوي على الإجراءات `procedure` والحزم

والدالات `functions` التي تقوم بكتابتها، ويتم استدعاء هذه

الوحدات من قبل وحدات أخرى ضمن تطبيقك.

(d) وحدات المكتبة الغرضية `object library modules`: وتسمح لنا

بتخزين العناصر التي يستخدمها تطبيق ما، وإعادة استخدامها في أي وقت تحتاجها

فيه، مما يساعدنا كثير في عمليات تطوير البرمجيات.

4-2 العناصر الرئيسية في وحدة النموذج form module

يرى معظم المستخدمون نموذج إدخال البيانات كعنصر وحيد، وكمبرمج عليك النظر إلى مكونات

النموذج بشكل أكثر تفصيلاً، وليس هذا فحسب بل كبنية هرمية، وفي oracle developer

تحتوي وحدة النموذج `form module` على العديد من العناصر الكبيرة والتي تحتوي بدورها

على عناصر أصغر. وتجمع كل هذه العناصر ضمن واجهة وحيدة متغاشكة تسمح للمستخدم بالتعامل

مع بيانات قاعدة البيانات، وفيما يلي توضيح لهذه المكونات وذلك بالترتيب من الأكبر إلى الأصغر

.

1-4-2 عناصر النافذة Window object

وهي الحدود الخارجية للنموذج، وهي عبارة عن إطار فارغ لحمل العناصر، ويتم تضمين جميع العناصر المرئية لتطبيق منشئ النماذج داخل النوافذ، ويوجد نوعين من النماذج وهما

.Document Windows (a) النوافذ المستديرة

.Dialog windows (b) النوافذ الحوارية

1-4-2 عناصر النافذة Window object

1-1-4-2 النوافذ المستديرة document windows

تستخدم النوافذ المستديرة ضمن نماذج إدخال البيانات، فإذا كنت تقوم بالتطوير في بيئة windows، سيحتوي تطبيقك تلقائياً على نوع من النوافذ يسمى نافذة واجهة المستند المتعددة Multiple Document Interface (MDI) والتي تمثل النافذة الأم التي تحتوي على كل النوافذ المستديرة الأخرى.

1-2-4-2 النوافذ الحوارية Dialog Window

تستخدم النوافذ الحوارية الرسائل للمستخدم، وعند تحديد خصائص نافذة حوارية يمكنك تحديد حجمها وموقعها، وأيضاً إذا كانت هذه النافذة الحوارية مشروطة modal أم لا فعلي سبيل المثال قد تجبر المستخدم من الإجابة عليها قبل أن يكون لديه القدرة على عمل شيء آخر، وكذلك تحديد غيماً إذا كان باستطاعة المستخدم نقلها أو تغيير حجمها أو تكبيرها أو تصغيرها أو إغلاقها وكذلك تحدي visual attributes كاللون والنقش ونوع الخط وحجمه وغيرها.

1-3-4-2 عناصر الكنافا أو البساط canvas object

يمكن أن تحتوي النافذة على كنافا واحدة أو أكثر، والكنافا عبارة عن الخلفيات المرئية visual background التي يتم وضع عناصر النموذج عليها، وهناك أربعة أنواع من

الكنافا أو البساط وهي المحتوى content والمكدة stacked والمبوبة tab، وشريط الأدوات toolbar.

content canvases 1-3-1-4-2

وتعتبر الخفية الأساسية لأي نموذج، ولذلك يتم وضع كنافا المحتوى تلقائياً داخل أي تخطيط نموذج جديد تقوم بإنشائه، بمعنى أنها النمط الافتراضي

stacked canvases 2-3-1-4-2

تسقى الكنافا المكدة فوق كنافا المحتويات content canvas مما يؤدي إلى إخفاء جزء منها، ويمكن أن تظهر الكنافا المكدة بنفس خفية كنافا المحتوى، أو شكل مختلف عنها، ويمكننا استخدام الكنافا المكدة لجعل العناصر تختفي أو تظهر على الشاشة اعتماداً على معيار أو شرط محدد.

tab canvases 2-3-1-4-2

وهي عبارة عن كنافا تحتوي على صفحات متعددة مفيدة للمستخدمين الذين يحتاجون إلى تحديد خيارات كثيرة ضمن النافذة windows، وهذا النوع يسمح للمستخدم بالانتقال بين صفحات متعددة ملزمة مترتبة بينها، وتكون مفيدة جداً عندما تريد إظهار كمية كبيرة جداً من المعلومات المرتبطة وذلك على مساحة صغيرة نسبياً، وتسقى الكنافا المبوبة على كنا المحتوى مما يؤدي إلى إخفاء جزء منها content canvas.

toolbar canvases 2-3-1-4-2

وهي نوعين أفقية horizontal، عمودية vertical، ويمكن للنافذة أن تحتوي شريط أو أكثر، ويمكنك التحكم بالكود بتحديد أشرطة الأدوات التي ستظهر في وقت معين، وكذلك تحديد العناصر المرئية ضمن كل شريط أدوات.

block object 3-1-4-2

الكتلة في منشىء النماذج عبارة عن حاوية منطقية لعناصر النموذج، مثل عناصر البيانات data item وأزرار التحكم control button، وهناك نوعان من الكتل وهي كتل البيانات data block، وكتل التحكم control block.

data block 1-3-1-4-2

تقوم كتل البيانات بإنشاء جسر بين نموذجك والبيانات الموجودة في قاعدة البيانات، وهي تسمح لك بالوصول إلى البيانات في جدول قاعدة البيانات data base table أو مشهد قاعدة بيانات data base view أو إجراء procedure، ويقوم منشىء النماذج بوضع إطار frame حول كتل البيانات، حيث يمكنك استخدام هذا الإطار لنقل الكتلة على البساط أو لاختيار الكتلة من تعديلها.

control blocks 2-3-1-4-2

تحتوي هذه الكتل عناصر لا تتفاعل مباشرة مع بيانات قاعدة البيانات، وإنما تحكم في التطبيق نفسه مثل الأزرار تقوم بوضعها ضمن كتلة تحكم.

item object أو البنود 4-1-4-2

العناصر items عبارة عن كائنات مستقلة موجودة ضمن نموذجك ويقوم المستخدم بالتفاعل معها، وأكثرها شيوعاً هو عنصر النص text item الذي يحتوي حقول تسمح للمستخدم ببرؤية وإدخال وتعديل بيانات قاعدة البيانات، ويوجد أيضاً عناصر لإظهار بيانات قراءة فقط مثل عنصر مخطط chart item وعنصر الإظهار display item، وعنصر الصور image item وغيرها من العناصر التي سوف نتعرض لها بالتفصيل لاحقاً.

3- إنشاء النماذج باستخدام معالج الكتل Data Block Wizard

يتمثل ذلك في الخطوات التالية:

(1) نقوم أولاً بتشغيل منشئ النماذج Form Builder، فيظهر صندوق الترحيب

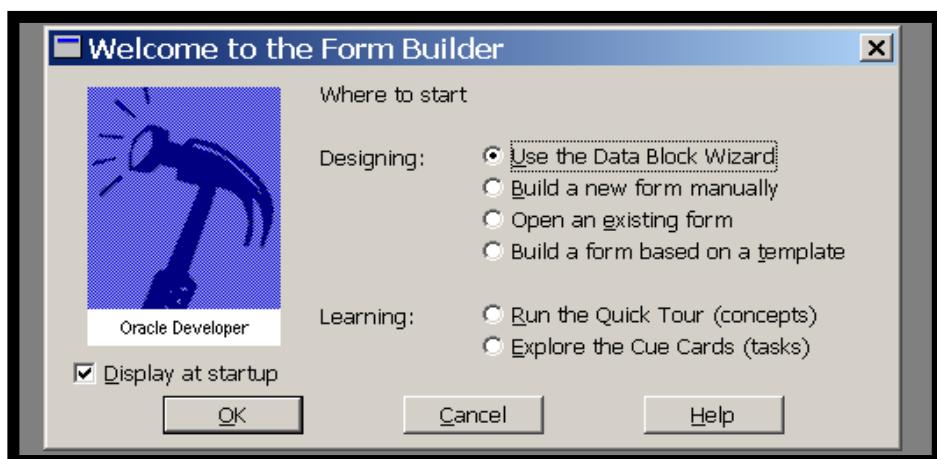
التالي فنختار الخيار use the data block wizard، ويساعدنا هذا

الخيار على إنشاء نموذج بطريقة سهلة حيث يساعدنا على الربط بين النموذج وقاعدة

البيانات، أو بشكل أدق بين النموذج ومصدر البيانات data source وذلك من

الجدول tables، والمشاهد views، والإجراءات procedures وغيرها

وسنلقي الضوء على هذه المصادر في وقت لاحق بالتفصيل ثم الضغط على ok.

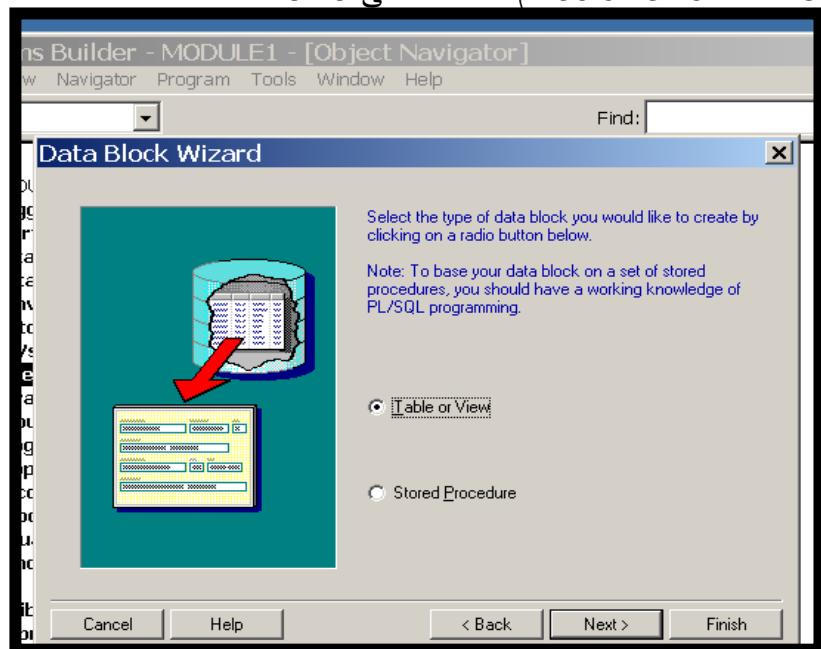


(2) تظهر لك شاشة الترحيب قم بالضغط على next .

3) تظهر لك الشاشة التالية تقوم بسؤالك عما إذا كانت بيانات الكتلة ستعتمد على جدول

أو مشهد view أو إجراء procedure، قم بتحديد الخيار table

next ثم الضغط على table or view



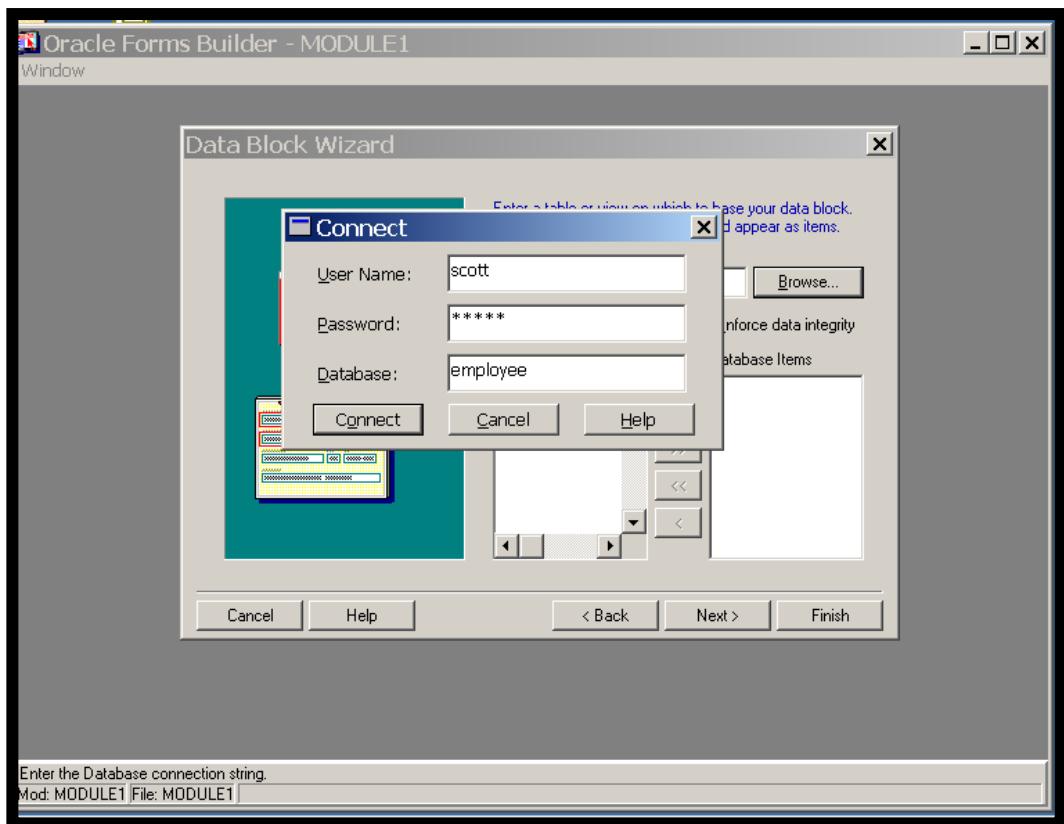
4) تظهر لنا النافذة التالية ويتم منها تحديد إسم الجدول المراد إنشاء نموذج بناءً على بيانات ،ولفعل

مثل ذلك لابد أولاً من الاتصال على قاعدة البيانات التي يعتبر هذا الجدول أحد مكوناتها ، وبالضغط

على استعراض browse وذلك لتحديد قاعدة البيانات مصدر البيانات data source ومن

ثم تحديد إسم الجدول مصدر البيانات data source فيقوم المعالج بالسؤال عن كلاً من إسم

المستخدم وكلمة السر واسم قاعدة البيانات فيتم إدخال تلك البند ثم الضغط connect.



ومن النافذة التالية يتم تحديد إسم الجدول وهو في مثالنا جدول emp، ثم الضغط على ok (5)

وتوجد بالنافذة عدة خيارات وهي:

Current User (a): للاتصال علي المستخدم الحالي وتحديد مصادر بيانات الكتلة

من جداول وغيرها.

Other User (b): للاتصال علي مستخدم آخر غير المستخدم الذي نحن بصدد

الاتصال عليه حالياً وتحديد مصادر بيانات الكتلة من جداول وغيرها.

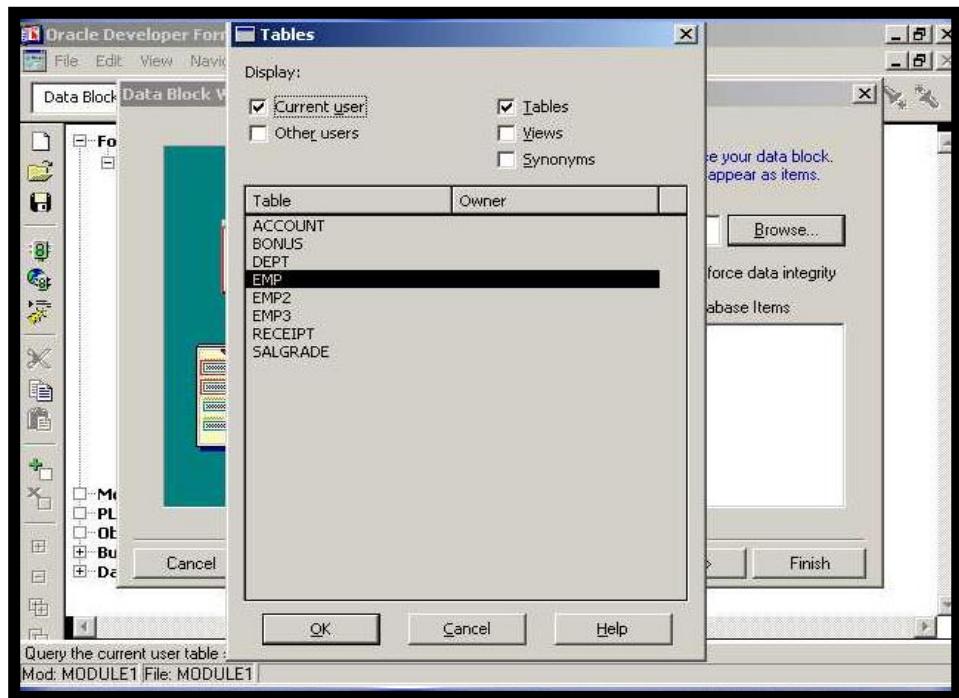
Tables (c): عند تنشيط هذا الخيار وجعله checked سيتم استعراض الجداول

الموجودة في قاعدة بيانات المستخدم الذي تم الاتصال عليه في الجزء الأسفل لتحديد أيًّا

منها سيكون مصدر بيانات الكتلة

Views (d) : عند تنشيط هذا الخيار وجعله checked سيتم استعراض المشاهد الموجودة في قاعدة بيانات المستخدم الذي تم الاتصال عليه في الجزء الأسفل لتحديد أي منها سيكون مصدر بيانات الكتلة.

Synonyms (e) : عند تنشيط هذا الخيار وجعله checked سيتم استعراض المترادفات الموجودة في قاعدة بيانات المستخدم الذي تم الاتصال عليه في الجزء الأسفل لتحديد أي منها سيكون مصدر بيانات الكتلة.



6) يتم ظهور أعمدة الجدول الذي تم اختياره في الخطة السابقة في قسم الأعمدة المتاحة

قم بإضافة كافة الأعمدة عن طريق الزر الذي يحتوي

إشارتي أكبر >> وذلك لنقل أسماء جميع الأعمدة من منطقة available

columns إلى منطقة database items، ثم قم بعد ذلك بالنقر على زر

next لظهور لك النافذة الأخيرة وبها خيارات وهو كال التالي:

Create the data block, then call the lay out wizard (a)

والذي يعني إنشاء الكتلة ثم استدعاء معالج التخطيطات وذلك للتحكم في إظهار أو إخفاء

.items

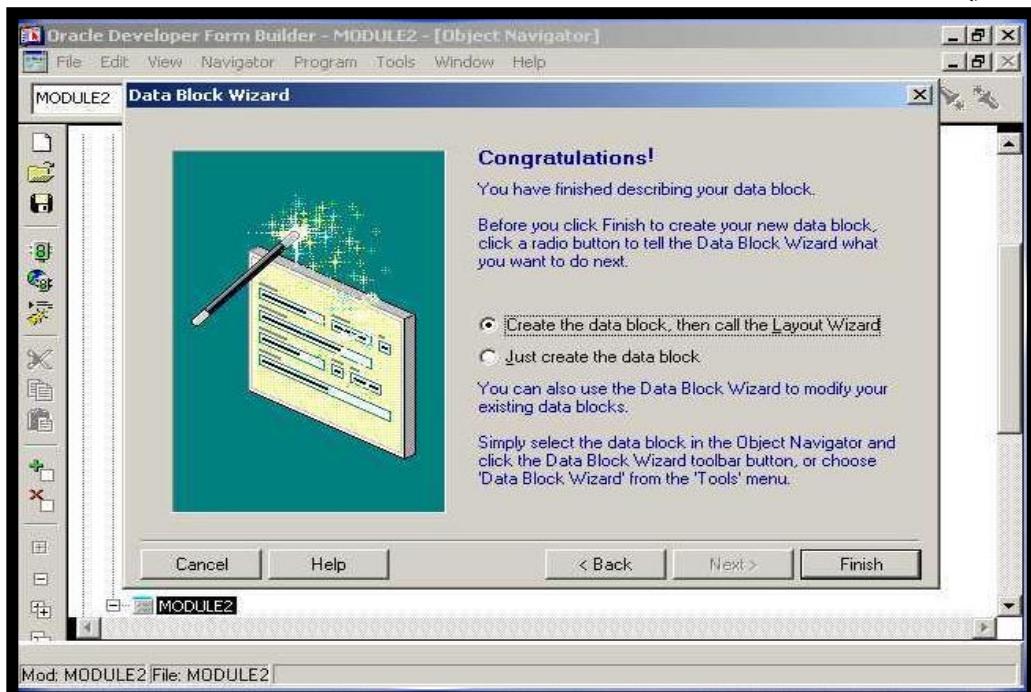
Just create the data block (b)

إنشاء كتلة المعطيات

(7) قم بتنشيط الخيار

Create the data block, and then call the lay out wizard

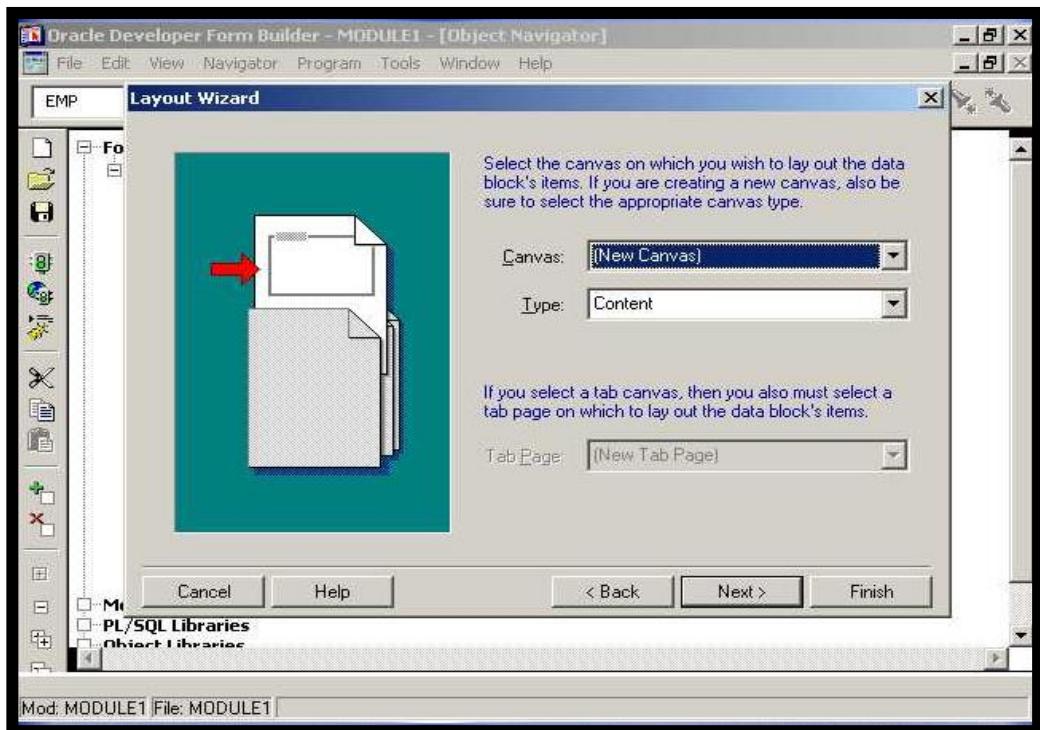
(8) النقر على الزر finish



(9) تظهر شاشة ترحيب معالج التخطيطات التالية قم بالنقر على زر next .

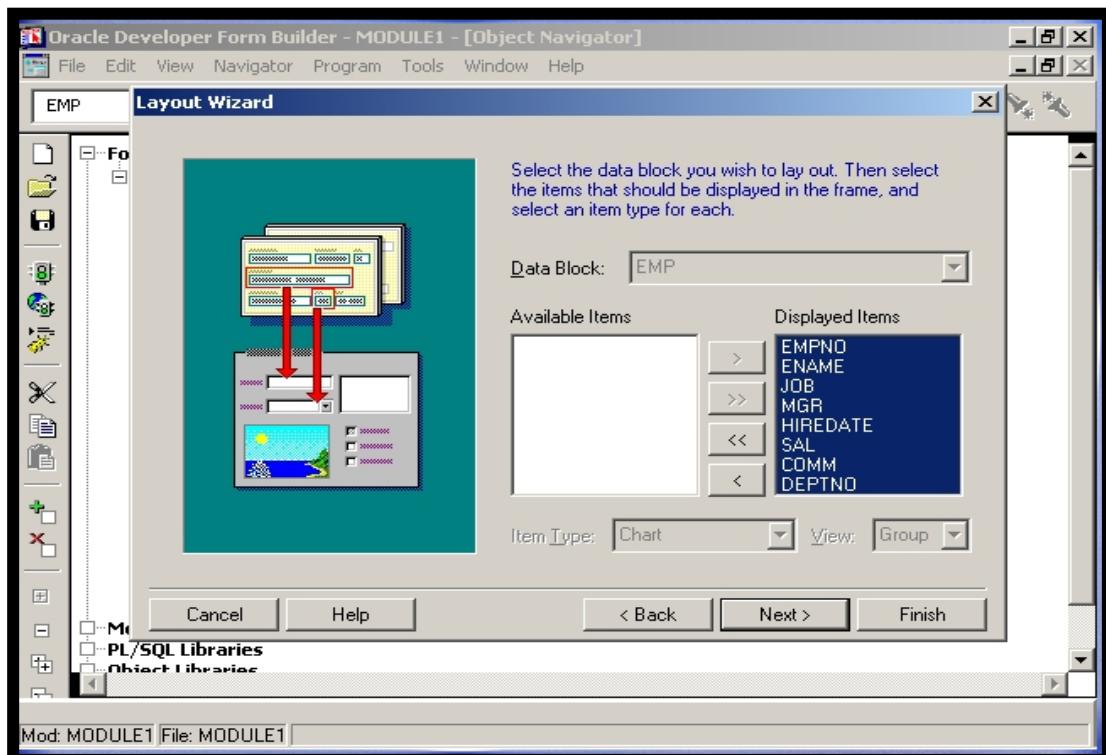


10) تظهر لنا شاشة الكنافا التالية التي تمكنا من تحديد إسم الكنافا ونوعها، حيث تقوم بسؤالك عن إسم الكنافا التي تريد استخدامها كما تعلمون أن الكنافا تعتبر الخلفية الأساسية لأي نموذج، ولذلك يتم وضع كنافا المحتوى تلقائياً داخل أي تخطيط نموذج جديد تقوم بإنشائه، بمعنى أنها النمط الافتراضي، قم بتحديد إسم الكنافا التي تريد ظهور الكتلة عليها ولأنك لم تقم بإنشاء كنافا سابقاً بمعنى أن هذا النموذج هو أول نموذج لك في تطبيقك فلن تظهر لك أية كنافا، فقم باختيار new canvas من type، ثم النقر على زر .next

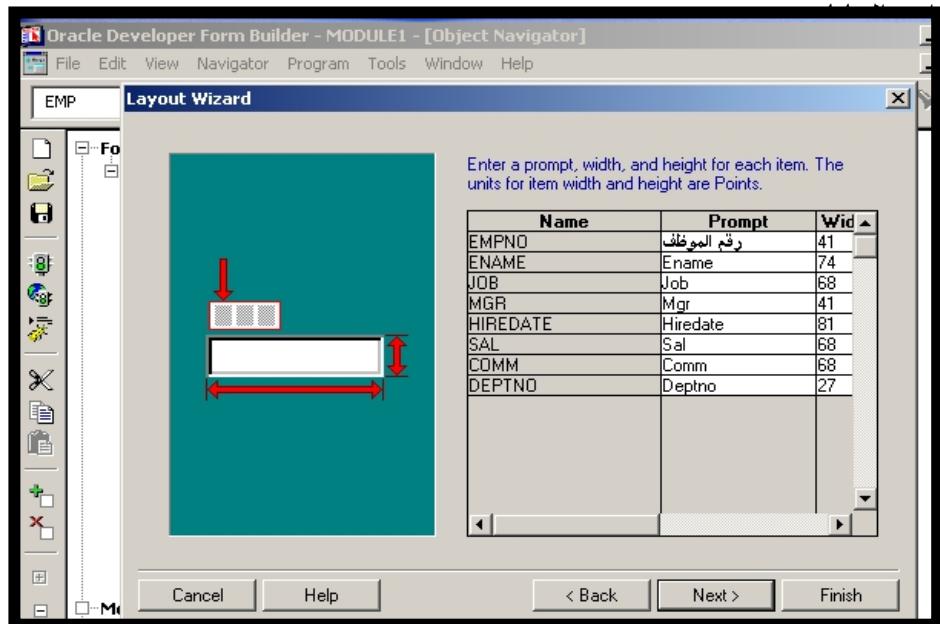


11) تظهر لنا نافذة كتلة البيانات التي تمكنا من التحكم في إظهار البنود علي الكتلة data displayed وذلك من خلال الضغط على الزر < إضافة عنصر أو الزر >> لإضافة كل العناصر block item

أو التراجع عن إظهار عنصر وذلك بإضافتها إلى قسم available item أو التراجع عن إظهار مجموعة العناصر عن طريق الزر .next.

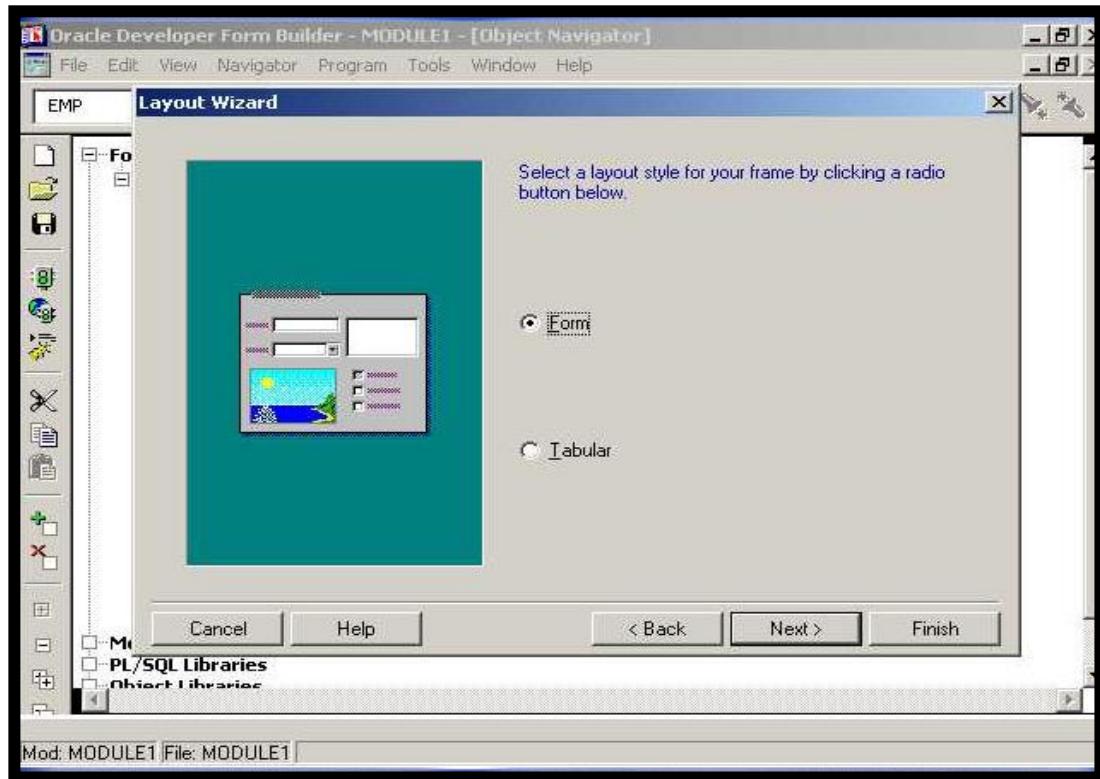


(12) تظهر لنا نافذة التحكم في العناصر التالية، ويمكنك من خلالها تغيير عنوان البدن من عمود prompt، وفي هذا المثال قمنا بتغيير العنوان label لبند empno ليصبح رقم الموظف ويمكنك التحكم في عرض وارتفاع البدن علي الكتلة وذلك من height، width، ثم قم بالضغط على زر .next

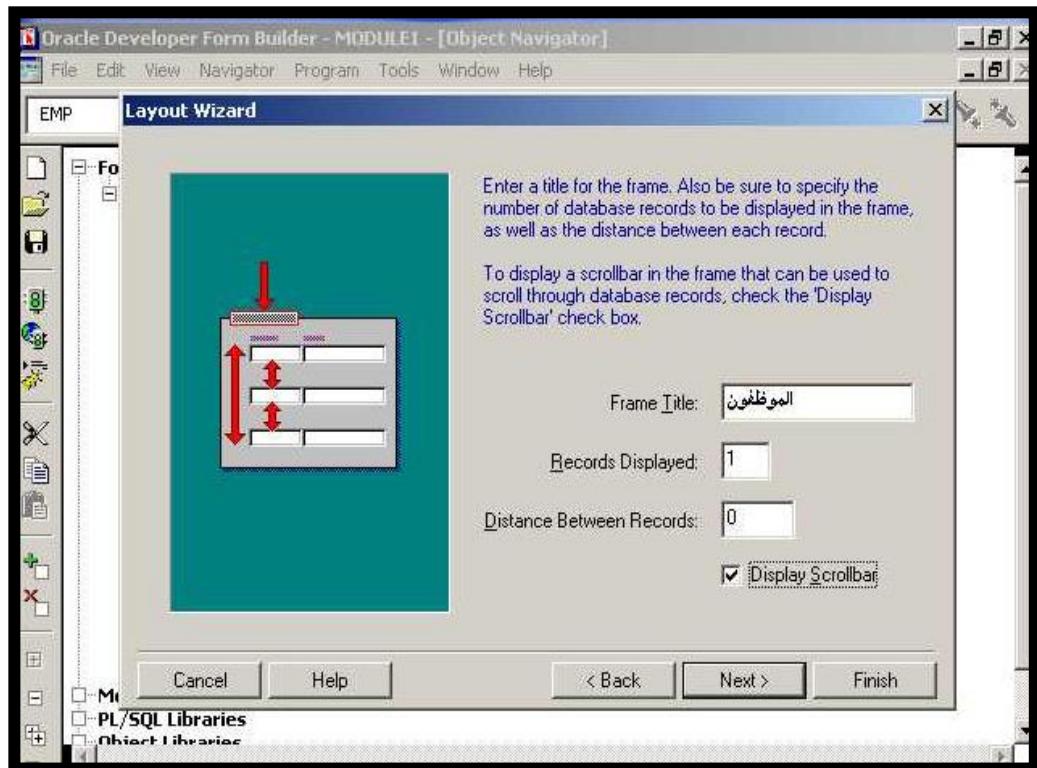


13) تظهر لنا نافذة النمط style layout حيث تقوم من خلالها بتحديد إذا ما كانت

بياناتك ستظهر وفق تخطيط نموذج form ، أو وفق تخطيط جدول table .next الضغط على form

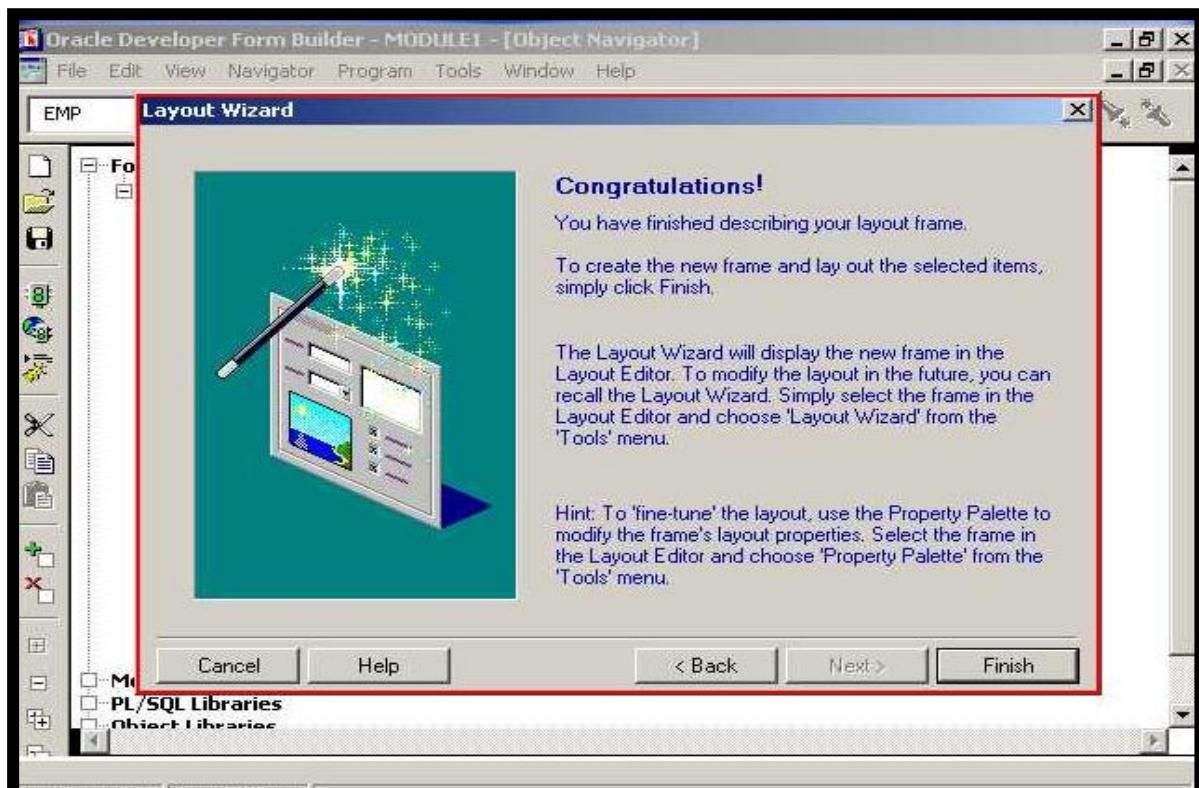


14) تظهر لنا النافذة التالية ونستطيع من خلالها إدخال عنوان للإطار من `frame title` والذي سيقوم معالج التخطيطات بوضعه حول عناصر البيانات وهو في هذا المثال الموظفون، ويسمح لنا بتحديد عدد السجلات التي ستظهر في وقت معاينة التخطيط وذلك من الحقل `records`، ولكننا قمنا باختيار النمط `form displayed`، فسنقوم بجعل عدد السجلات 1، ويمكننا تحديد المسافات بين السجلات وذلك من الحقل `distance between record` والأفضل ترك القيمة الافتراضية لهذا الحقل وهي 0 حتى لو ظهرت السجلات قريبة من بعضها حيث بإمكانك زيادة المسافة بينها من خلال محرر التخطيطات `layout editor` والذي يمكننا من توزيع العناصر بشكل متساوٍ وإظهار النتيجة مباشرة مما يساعد على تحديد المسافات بشكل أفضل وأسرع، أما الحقل الأخير `display scrollbar` يمكننا من تحديد ما إذا كان الإطار سيحتوي شريط تمرير أم لا ويكون ذلك مفيداً عندما يعرض الإطار أكثر من سجل، حدد ما تريده ثم قم بالضغط على زر `next`.

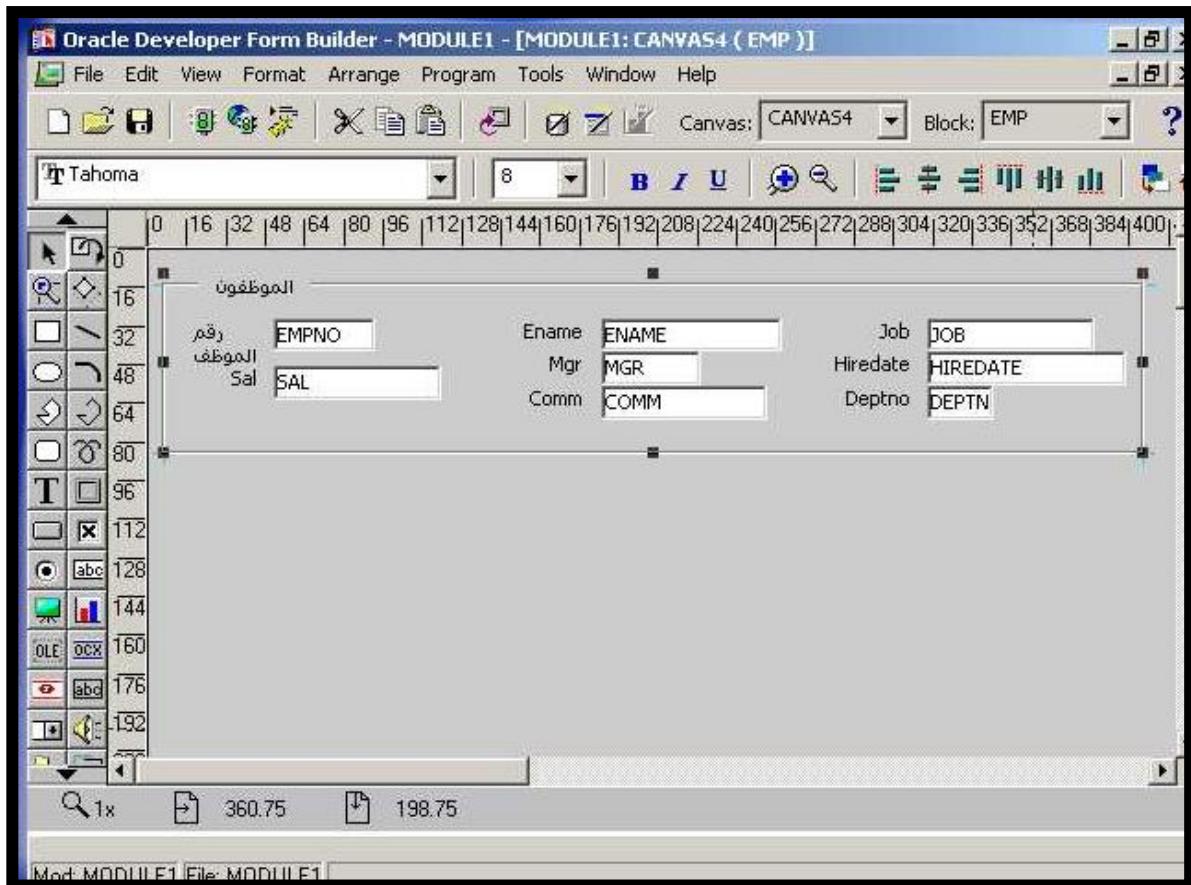


15) يقوم معالج التخطيطات بإظهار نافذة التهنة الأخيرة التالية قم بقراءتها ثم أنقر الزر

.finish



(16) يقوم معالج النماذج form builder بإنشاء نموذجك وإظهاره في محرر التخطيطات وذلك كما بالشكل التالي، يمكنك تشغيل معالج التخطيطات layout editor مرة أخرى على تخطيط موجود مسبقاً، وذلك لإجراء تغييرات عامة على هذا التخطيط بشكل سريع، وكما يمكنك أيضاً تشغيل معالج التخطيطات layout wizard من خلال محرر التخطيطات layout editor بالنقر على الزر  في شريط الأدوات.



4 - حفظ وترجمة وتشغيل وحدة نموذج :form module

بإمكانك إنشاء وحدات نموذج في منشئ النماذج لكنك لن تستطيع تشغيلها فيه، ومن أجل ذلك عليك تشغيل برنامج تنفيذ النماذج Form Runtime قبل القيام بذلك يجب عليك القيام بحفظ النموذج ثم ترجمته.

4-1 حفظ وحدة نموذج :saving a form module

ويكون ذلك بإحدى الطريقتين التاليتين:

(a) قم بتنفيذ الأمر save من قائمة file | save أو النقر على زر save مباشرة



والذي يأخذ الشكل المقابل

(b) توصيف منشئ النماذج form builder للقيام بحفظ عملك تلقائياً في

كل مرة تقوم فيها بترجمته وذلك من خلال تنفيذ الأمر

حدد الخيار general tools | preferences ثم من تبويب

Save before Building

2-4 ترجمة وحدة نموذج :compiling a form module

لا يمكنك تشغيل وحدات النماذج التي قمت بإنشائها في منشئ النماذج form Builder

مباشرة، وإنما توجد خطوة فاصلة يتم تنفيذها لتحويل ملف وحدة النموذج إلى ملف تنفيذي منفصل

حيث يستطيع برنامج تنفيذ النماذج Form Runtime قراءته، ويسمى هذا الإجراء بالترجمة

، ويمكنك ترجمة وحدات النماذج في أي وقت بتتنفيذ أمر القائمة compiling

File | Administration | Compile File

يمكنك توصيف منشئ النماذج Form Builder كي يقوم وبشكل تلقائي

ترجمة وحدة النموذج قبل تنفيذها، قم بتتنفيذ أمر القائمة Tools | preferences

وفي تبويب General حدد الخيار Build Before Running

تمرين

3-4 تنفيذ وحدة نموذج :Running a form module

للقيام بتتنفيذ وحدة نموذج، قم بتتنفيذ الأمر Program | Run Form أو النقر على الزر



والذي يشبه علامة المرور ، والذي يأخذ الشكل المقابل Run form

- ✓ الشيء الأول الذي تلاحظه عند تشغيل النموذج هو أن النموذج لا يقوم بإظهار

أية بيانات من قاعدة البيانات لأن برنامج تنفيذ النماذج Form Builder

يفترض بأنك عندما تقوم بفتح النموذج لأول مره فستحتاجه لإدخال بيانات جديدة، أما

إذا أردت إظهار البيانات فيجب عليك إجراء الاستعلام عليها وذلك بتنفيذ الأمر

Execute Query أو النقر مباشرة على زر Query | Execute

4-4 أشكال ملفات منشئ النماذج ومميزاتها:

بعد أن قمنا بالإلمام بعمليات إنشاء وترجمة وتنفيذ وحدة نموذج، يجب علينا الإشارة إلى

الأنماط المختلفة لملفات منشئ النماذج Form builder، حيث يقوم كل مكون من

مكونات منشئ النماذج بإنشاء مجموعة مختلفة من الملفات، وكل ملف منها هدف مختلف

عن الآخر، فمكون النموذج form يقوم بتوليد ثلاثة أنماط من الملفات، فملف التصميم

الذي تقوم بالتعامل معه مباشرة يأخذ الامتداد fmb (اختصارا للعبارة

(form module binary) ، أما الملف التنفيذي الذي يتولد عندما تقوم بترجمة وحدة

النموذج فيأخذ الامتداد fmx (form module executable).

والجدول التالي بين المكون والملفات التي يكونها في وضع التصميم والتنفيذ وامتداد كل

منها.

الوصف	الامتداد	الوحدة أو المكون
ملف تصميم النموذج	.fmb	نموذج Form
ملف تنفيذ النموذج	.fmx	
ملف تصميم القائمة	.mmx	قائمة menu
ملف تنفيذ القائمة	.mmx	
ملف تصميم مكتبة PL/SQL ويمكن تنفيذه أيضاً يحتوي على الترميزين المصدر والتتنفيذ	.pll	مكتبة PL/SQL
ملف تنفيذ مكتبة PL/SQL ولا يحتوي على أي ترميز مصدر.	.plx	
ملف تصميم مكتبة العناصر ملف التصدير النصي لمكتبة العناصر.	.olb .olt	مكتبة العناصر object library

5- إدراج وتعديل وحذف السجلات :Records

لكي نقوم بعمليات إضافة السجلات، وتعديلها بـتغيير محتوياتها، وحذفها، وقد قمنا بذلك

من قبل باستخدام أمر لغة Data Manipulation Language ولا يتم

تنفيذها على قاعدة البيانات الأساسية حتى تقوم بعملية التثبيت Commit لهذه التغييرات

بـاستخدام أمر save.

1-5 إدراج السجلات :Inserting Records

يمكنك إدراج السجلات بإحدى الطرق التالية:

(a) اختر أمر القائمة Record | insert

(b) أنقر على زر إدراج السجلات Insert Record الموجود في شريط الأدوات

(c) الانتقال إلى السجل الأخير ومن ثم الضغط على مفتاح السهم السفلي Down Arrow للانتقال إلى سجل جديد.

5-2 تعديل السجلات :updating Records

إن إجراء التعديل من خلال النماذج عملية سهلة وبسيطة، فما عليك سوى الانتقال إلى السجلات المطلوبة، وانقل المؤشر إلى الحقل الذي تريده تغييره وتعديلاته، وقم بإجراء التعديل المطلوب، ثم قم بعملية الحفظ.

5-3 حذف السجلات :Deleting Records

إن عملية حذف السجلات من خلال النماذج عملية سهلة وبسيطة، وهي أسهل من تعديلها وتتم بإحدى الطرق التالية:

(a) الانتقال أولاً إلى أي حقل في السجل المراد المطلوب حذفها ثم القيام بتنفيذ أمر

القائمة Record | Remove.

(b) الانتقال أولاً إلى أي حقل في السجل المراد المطلوب حذفها ثم القيام بالنقر على

زر Remove Record الموجود في شريط الأدوات.

(c) الانتقال أولاً إلى أي حقل في السجل المراد المطلوب حذفها ثم القيام بضغط

مفاتحي shift+f6 من لوحة المفاتيح.

لا تنس القيام بعملية التثبيت commit وذلك بالقيام بالضغط على زر save من شريط الأدوات.

5-4 فهم نمطي عمليات تنفيذ النماذج:

يكون للنموذج الفعال في Form Runtime إحدى حالتين:

الأولي: الحالة العادية normal mode وهي حالة الإدراج والتعديل والحذف

الثانية: حالة الاستعلام Query mode وهي حالة الاستعلام ويقوم المستخدم باستدعاء هذا النمط بتنفيذ الأمر execute Query يقوم النموذج بتغريغ محتواه وانتظار المستخدم القيام بإدخال معيار تصفية Filtering Criteria لعبارة Where

5-5 إضافة الكائنات إلى النموذج:

(1) لإضافة عناصر النص يتم الضغط على رمز عنصر النص من شريط أو مربع الأدوات البرمجية في أقصي اليسار ثم تحديد حجمه على النموذج ومن ثم تحديد مجموعة



الخصائص الخاصة به والذي يأخذ هذا الشكل

(2) لإضافة عناصر الإظهار item Display يتم الضغط على رمزه من شريط أو مربع

الأدوات البرمجية في أقصي اليسار ثم تحديد حجمه على النموذج ومن ثم تحديد مجموعة،

وعنصر الإظهار هو عبارة عن عنصر نموذج يقوم بإظهار البيانات للمستخدم إلا أنه لا



يسمح بإدخالها أو التعديل فيها ويأخذ هذا الشكل

(3) بالإضافة للعناصر التي تساعد المستخدم على إدخال البيانات وتعديلها يحتاج النموذج إلى

عناصر تقوم بإظهار بيانات للقراءة فقط منها أو أزرار يتم إضافة بنفس الطريقة إلى

النموذج في وضع التصميم منها الزر Radio Button الذي يشبه الشكل التالي



وكذلك Check box

(4) وبنفس الطريقة يمكننا إضافة عناصر الصورة والصوت وغيرها إلى النموذج.

6- تحديد وتعديل الخصائص للكائنات الموجودة في النموذج

سنتعرف في هذا الجزء على مكونات لوحة الخصائص Property Palette والتي يتم

من خلالها تحديد كيفية ظهور كل عنصر من عناصر النموذج وعمل هذا العنصر، وتساعدك

لوحة الخصائص كثيراً في تطوير النماذج كالخصائص التي تتحكم بظاهر وسلوك كتل البيانات

وكذلك خصائص كلامن الإطارات وكتل التحكم، وستتعرف أيضاً على خصائص العناصر النصية التي تسمح لك ببرؤية وتحرير النصوص والأرقام والتاريخ وغيرها من البيانات وكيفية إجبار المستخدم على إدخال بيانات بشكل معين وكيفية إنشاء أدوات الإيضاخ

hints واللمحات Tool tips

1-6 معالجة الخصائص Manipulating Properties

تحتوي لوحة الخصائص على أنواع متعددة من حقول القيم وهي موضحة كالتالي:

a) حقول حرفية رقمية Alphanumeric لكتابة القيم.

b) حقول تحتوي على زر بالعنوان More، وهي تفتح صندوق حوار لتحديد القيم.

c) حقول مع زر وهي تظهر علامة..... تقوم بفتح نافذة لإدخال وتحرير نصوص

.where طويلة كالتعليقات أو عبارات

d) قوائم مسدلة Drop-down list بقيمة ثابتة ومعرفة مسبقاً مثل حقول

change، yes/no أو خيارات نمط الإبخار لكتلة بيانات ثابتة مثل

record, same, change data block

وببساطة يمكنك تغيير قيم الخصائص وذلك بالنقر على حقل القيمة للخاصية المطلوبة، ثم كتابة القيمة الصحيحة أو تحديدها في صندوق حوار وهكذا... ويمكنك مقارنة خصائص عنصرين أو أكثر وذلك باختيار جميع العناصر التي نرغب بمقارنتها، ومن ثم استخدام عدة لوحات خصائص ، فإذا كنت ضمن نمط إظهار التقاطع intersection الافتراضي للوحة الخصائص، فستقوم اللوحة بإظهار الخصائص المشتركة فقط بين جميع العناصر المختارة، والخصائص التي

لها نفس القيمة في جميع العناصر المختارة فستظهر فيها القيمة المشتركة أما إذا لم تكن لها نفس

القيمة فسيظهر الرمز **** ضمن الخصائص بدلاً من القيمة

تميّز

✓ إذا أردت إظهار خصائص لعدة عناصر في لوحة وحيدة، وبالتالي فإن أيه

خاصية ستتغير في جميع العناصر المختارة.

✓ يمكنك اختيار عدة عناصر من مستكشف العناصر الموضح في الشكل التالي

بإستخدام نفس التقنيات التي تتعامل بها مع مستكشف النوافذ windows

shift explorer، وبعد اختيار العنصر الأول وبالضغط على مفتاح

✓ مع النقر على عنصر آخر يؤدي إلى اختيار مجال متلاصق يحتوي على جميع

العناصر بين الاختيار الأول والاختيار الثاني، أما إذا أردت اختيار عناصر غير

متجاورة استخدم مفتاح `ctrl` مع النقر على العناصر المحددة.

✓ وإظهار نافذة خصائص العنصر نقوم بتحديد العنصر ثم الضغط مرتين

`f4` أو تحديد العنصر ثم الضغط على Double click

6-2 خصائص كتلة البيانات **data block**

تنقسم خصائص كتلة البيانات إلى تسعة فئات وهي: عامة general، وإبحار navigation

والسجلات record، وقاعدة البيانات database، قضيب الانزلاق scrollbar، والبنط واللون

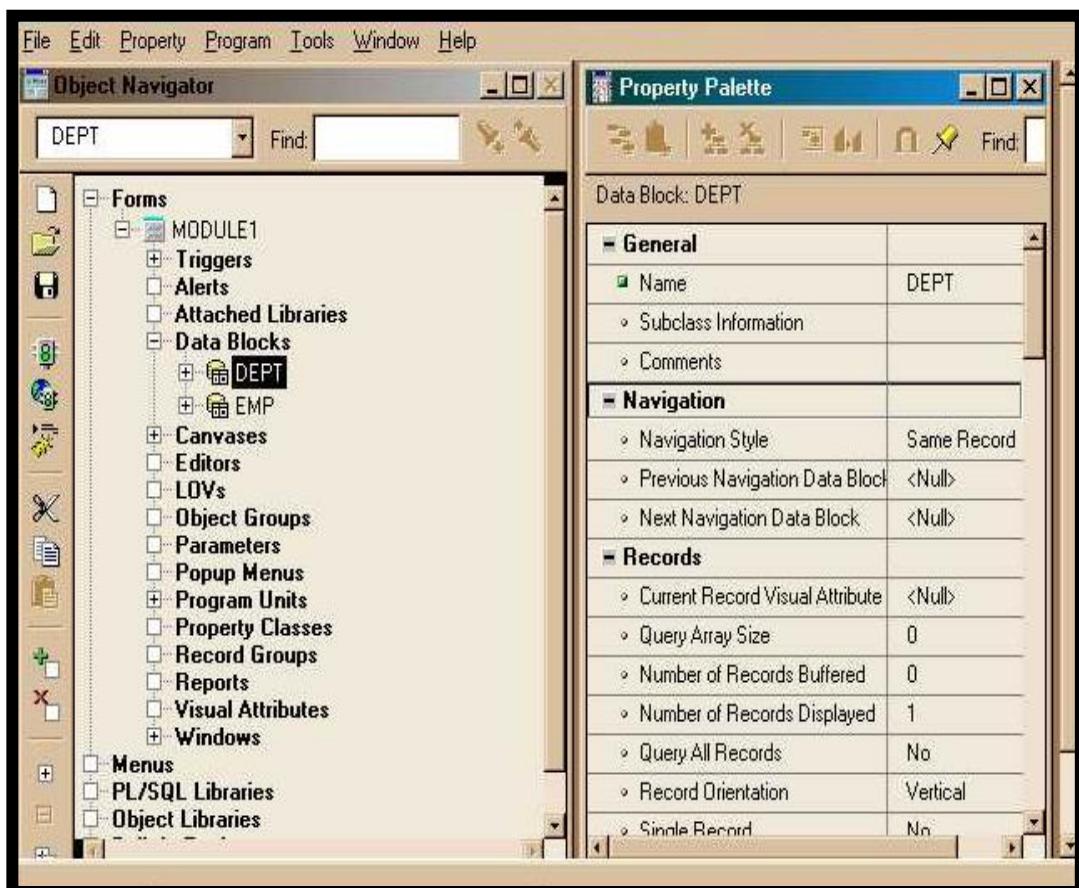
international character mode ونمط الحروف font & color

وقاعدة بيانات متقدمة advanced database، وإظهار نافذة خصائص كتل البيانات نقوم

بتحديد الكتلة وهي في مثالنا emp أو تحديد العنصر ثم Double click ثم الضغط عليها مرتين

الضغط على F4

فظهور نافذة الخصائص في الركن الأيمن من الشاشة كما بالشكل التالي:



والجدول التالي يوضح أهم خصائص كتلة البيانات:-

Function الوظيفة	property name إسم الخاصية
ولها خيارات yes, no وفهي حالة اختيار yes نحدد أن الكتلة مبنية على جدول أو كائن قاعدة البيانات	Data base data block
التحكم بعدد السجلات التي يمكن استرجاعها من قاعدة البيانات في وقت ما	Query array size
نحدد منها إسم الكائن أو الجدول مصدر البيانات الذي عند الاستعلام سيتم استرجاع البيانات منه	Query data source name
عدد السجلات التي يمكن لـ form الاحتفاظ بها في ذاكرة الحاسب client	Number of records buffered
خاصية القيمة الافتراضية لعدد السجلات الظاهرة	Number of records displayed
لوضع شرط أو عدة شروط	Where clause
ولها خيارات yes, no وفهي حالة اختيار yes يمكن للمستخدم إدخال معيار تصفية لقييد السجلات المسترجعة	Query allowed
تسمح لنا بتحديد كيفية فرز السجلات المسترجعة	Order by clause
تمكننا من التحكم بعمليات الإدراج والتعديل والحذف ضمن كتلة المعطيات	Insert/update/delete allowed

3- التحكم في خصائص الإطارات frame properties

هناك عدة خصائص قد تكون لها أهمية خاصة عند التعامل مع الإطارات على الكنافا في

تطبيقك كما هو موضح بالجدول التالي:

الوظيفة function	إسم الخاصية property name
التحكم بعدد السجلات التي يمكن إسترجاعها من قاعدة البيانات في وقت ما	Frame title alignment
خاصية القيمة الإفتراضية لعدد السجلات الظاهرة في الإطار	Number of records displayed
وتحدد شكل بنود الإطار وتحتوي خاصيتان الأولى form وتعني شكل نموذج والثانية tabular وتعني جولي	Layout style
تمكننا من تحديد المسافة بين السجلات المسترجعة	Distance between Record
تسمح لنا بتحديد كيفية فرز السجلات المسترجعة	Order by
تمكننا من التحكم بعمليات الإدراج والتعديل والحذف ضمن كتلة البيانات .	Insert/update/delete allowed

4- التحكم في خصائص عناصر النص text items

في هذا الجزء سنطرق إلى أهم جزء في أي نموذج ألا وهو عنصر النص TEXT ITEM ،

حيث يقوم منشيء الماژاج Form Builder باستخدام عناصر النص لإظهار نص أو تاريخ

أو أرقامإلخ وتسمح بتحرير البيانات من قبل المستخدم، ويجب الانتباه إلى أن أي عنصر

يختلف عن عناصر الإظهار Display item في أن عنصر الإظهار text item

لا يسمح بتغيير البيانات الظاهرة ، ويمكن لعنصر النص أن يحتوي سطرا واحدا أو عدة أسطر .

Text item appearance - 4 - 1 - تعديل مظهر عنصر النص

حيث أن الهدف الرئيسي من عناصر النص text items يتمثل في إظهار البيانات، فنجد أنه من الضروري الإلمام بكيفية التحكم بمظهر البيانات التي نقوم بإظهارها، ويمكنك الوصول إلى خصائص الإظهار باختيار العنصر الذي نرغب في تغيير مظهره ثم فتح لوحة خصائص العناصر property palette والجدول التالي يوضح أهم الخصائص المتعلقة بظاهر عنصر

النص:

Function الوظيفة	Property name إسم الخاصية	Category name إسم الفئة
محاذاة أفقية لمحتوى العنصر	justification	functional
تمكّنك من التحكم بإظهار وتتنسيق النصوص والأرقام والتواريخ	Format mask	data
التحكم في المسافة العمودية بين السجلات.	Distance between record	Records
إظهار أو إخفاء العنصر للمستخدم	visible	Physical
إن خاصية attribute group تسمح لنا بإنشاء مجموعة خصائص البنط والألوان والتعبئة زمن ثم تطبيق هذه الخصائص على نموذجك وهي بمثابة التنسيق التلقائي في برنامج معالجة النصوص	Visual attribute group	Font&color
عنوان نص يتم إظهاره مع العنصر	prompt	prompt

التحكم في بيانات عنصر النص - 4 - 2 - 6

إذا كان مظهر عنصر النص يعتبر أمر ضروري وهام إلا أن محتوى عنصر النص من البيانات

أكثـر أهمية والجدول التالي يوضح الخصائص التي تمكـنا من التحكم في البيانات التي يحتـويـها

عنـصر النـص:

Function الوظيفة	Property name إسم الخاصية	Category name إسم الفئة
تمكـنا من إدخـال عـدة أـسـطـر من الـبيانـات فـي عنـصر النـص	Multi-line	functional
لـقـب الـقيـم المـدخلـة من قـبـل المـسـتـخـدـم إـلـى أحـرـف كـبـيرـة أو صـغـيرـة .	Case restriction	functional
الـتـحـكـم فـي المسـافـة العمـودـية بـيـن السـجـلات.	Distance between record	Data
لتـحـدـيد نوع الـبيانـات التي يمـكـن لـعـنصر النـص قـبـولـها.	datatype	Data
الـحد الأقصـى المـمـكـن لـلـعـنصر قبـولـه	Maximum length	Data
تـقـبـل فـقـط الإـدخـالـات التي تحـتـوي عـلـى عـدـد الأـحـرـف المـحـدـدـة تـامـا فـي الخـاصـيـة fixed	Data fixed	Data
الـقـيـمة الـافتـراضـيـة لـلـعـنصر	Initial value	Data
مـطـلـوب وتحـتـوي خـيـارـان yes , no اختـيـار yes فـلـن يـسـمح لـمـسـتـخـدـم بـحـفـظ السـجـل إـلـا إـذـا احـتـوي العـنصر عـلـي قـيـمة.	required	Data
أـقـل قـيـمة مـقـبـولـة	Lowest value	Data

أكبر قيمة مقبولة	Highest allowed value	Data
لتحديد ما إذا كان سيتم حساب العنصر من خلال صيغة formula أو تلخيص summary	Calculation mode	calculation

6 - 4 - 3 تعديل سلوك الإبحار لحقل نص

بعد تحديد كلًّا من المظهر والتحكم في بيانات عناصر النص ، نقوم بالتركيز على الخصائص

التي تسهل عملية التنقل ضمن التطبيق كما هو موضح بالجدول التالي :

Function الوظيفة	Property name إسم الخاصية	Category name إسم الفئة
وتمكننا من تحديد أن المؤشر سيقفز إلى العنصر التالي بعد الإضافة أو تعديل الحرف الأخير	Automatic skip	functional
وتحتوي خيارات yes، no وفي حالة اختيار no فلن يتم نقل التركيز أو المؤشر إلى العنصر باستخدام مفاتيح التنقل	Keyboard navigable	Navigation
لتحديد العنصر الذي سيتم الانتقال إليه عندما يزيد المستخدم الرجوع للخلف.	Previous navigation item	Navigation
لتحديد العنصر الذي سيتم الانتقال إليه عندما يزيد المستخدم التقدم نحو الأمام.	Next navigation item	Navigation
إظهار أو إخفاء شريط التمرير	Show scroll bar	physical

4 - 4 - ربط عنصر نص بقاعدة البيانات

في هذه الفقرة سنحدد الخصائص التي تسمح لك بالتحكم في كيفية ربط عنصر نص بقاعدة البيانات، وهناك العديد من الخصائص التي تمكنا من تحديد ما هي العناصر التي يستطيع المستخدم من إضافة البيانات وتعديلها وتحت أية معايير أو شروط محددة وغيرها كما هو موضح بالجدول التالي :

Function الوظيفة	Property name إسم الخاصية	Category name إسم الفئة
تحديد خاصية المفتاح الأساسي لهذا العنصر	Primary key	database
وتحتوي خياران yes, no وفي حالة اختيار yes يمكننا إجراء الاستعلام على هذا العنصر فقط ولا يمكننا إجراء عمليات الإدراج أو التعديل على هذا العنصر	Query only	database
وتحتوي خياران yes, no وفي حالة اختيار yes يمكن للمستخدم إجراء الاستعلام .	Query allowed	database
وتحتوي خياران yes, no وفي حالة اختيار yes فإن أي إستعلام سيتم إنشائه من منشئ النماذج سيكون Case - insensitive	Case - insensitive query	database
وتحتوي خياران yes, no وفي حالة اختيار yes فإن المستخدم يستطيع إضافة	Insert allowed	database

البيانات إلى عناصر النصوص.		
وتحتوي خيارات yes, no وفي حالة اختيار yes فإن المستخدم يستطيع تعديل البيانات في عناصر النصوص.	update allowed	database
تسمح لنا بتحديد أنه يجب أن يقوم هذا العنصر بإستناد قيمته من صيغة أو تلخيص بدلاً من عمود قاعدة البيانات	Calculation mode	Calculation
لتخزين تعبير pl/sql لحساب قيمة عنصر وذلك إذا كان تم تحديد الخاصية السابقة على Formula	formula	Calculation
لتحديد الدالة المستخدمة لحساب قيمة العنصر مثل count, max, min, sum, average	Summary function	Calculation

4 - 5 تضمين رسائل مساعدة في تطبيقك

يزودنا منشيء النماذج بخصائص تجعل من السهل إضافة معلومات مساعدة لكي يراها المستخدم عند الانتقال من عنصر إلى آخر، ويمكن أن تأخذ هذه الخاصية أحد شكلين الأول: وهي التلميحات hints ويتم ظهرها في سطر الرسالة وذلك أسفل نافذة النموذج. الثاني: أدوات الإيضاح تظهر ملاصقة لمؤشر الفأرة وذلك عند وضعها لبرهة فوق العنصر والجدول التالي يوضح قائمة بهذه الخصائص:

Function الوظيفة	Property name إسم الخاصية	Category name إسم الفئة

<p>لتخزين نص التلميح الذي نرغب بعرضه علي المستخدمين عندما يقومون بالدخول إلى عنصر النص هذا</p>	hint	help
<p>لتخزين نص أداة الإيضاح الذي نرغب بإظهاره للمستخدم عندما يحوم مؤشر الماوس فوق عنصر النص بالدخول إلى عنصر النص هذا</p>	tooltip	help

الأسئلة

المجموعة الأولى:

ضع علامة (✓) أمام العبارة الصحيحة وعلامة (✗) أمام العبارة الخاطئة مع تصحيح الخطأ.

1) منشئ النماذج يساعدك على إنشاء شاشات إدخال البيانات والمعروفة بالنماذج.

2) منشئ التقارير عبارة عن أداة تستخدم لإنشاء التقارير ضمن تطبيقات

ويساعدك في إنشاء التقارير في مشروعك أو تطبيقك وربطها بالشاشات Oracle

المختلفة من خلال بيئة رسومية سهلة.

3) منشئ الاستعلامات يساعد المبرمجين على ترجمة نصوص تطبيق ما إلى لغات برمجة

أخرى

4) منشئ التقارير يساعد على إنشاء الرسوم البيانية المختلفة من خلال بيئة رسومية GUI.

5) عناصر النافذة هي الحدود الخارجية للنموذج، وهي عبارة عن إطار فارغ لحمل العناصر.

6) يمكن أن تحتوي النافذة على كنافا واحدة أو أكثر.

7) تقوم كتل البيانات بإنشاء جسر بين نموذجك والبيانات الموجودة في قاعدة البيانات.

8) يتم وضع الكنافا المكدسة تلقائياً داخل أي تخطيط نموذج جديد تقوم بإنشائه، بمعنى أنها

النمط الافتراضي.

9) تستقر الكنافا المكدسة فوق كنافا المحتويات content canvas مما يؤدي إلى

إخفاء جزء منها.

10) الكتلة في منشئ النماذج عبارة عن حاوية منطقية لعناصر النموذج.

المجموعة الثانية:

- 1) ماهي المكونات الرئيسية الأربع لمنشيء النماذج ؟
- 2) ماهي العلاقة بين النافذة الكنافا ؟
- 3) ماهي أنواع الكنافا مع المقارنة بينها ؟
- 4) حدد الطريقة التي يمكننا بها حذف عنصر من تخطيط النموذج دون حذفه من كتلة البيانات ؟