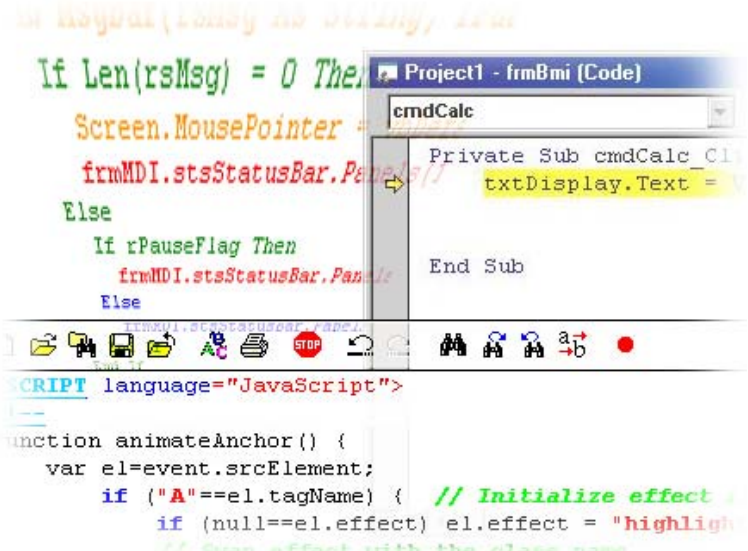
البرمجيات

مقدمة قواعد بيانات أوراكل

١٦١ حاب



مقدمة

الحمد لله وحده، والصلاة والسلام على من لا نبي بعده، محمد وعلى آله وصحبه، وبعد:

تسعى المؤسسة العامة للتعليم الفني والتدريب المهني لتأهيل الكوادر الوطنية المدربة القادرة على شغل الوظائف التقنية والفنية والمهنية المتوفرة في سوق العمل، ويأتي هذا الاهتمام نتيجة للتوجهات السديدة من لدن قادة هذا الوطن التي تصب في مجملها نحو إيجاد وطن متكامل يعتمد ذاتياً على موارده وعلى قوة شبابه المسلح بالعلم والإيمان من أجل الاستمرار قدماً في دفع عجلة التقدم التنموي؛ لتصل بعون الله تعالى لمصاف الدول المتقدمة صناعياً.

وقد خطت الإدارة العامة لتصميم وتطوير المناهج خطوة إيجابية تتفق مع التجارب الدولية المتقدمة في بناء البرامج التدريبية، وفق أساليب علمية حديثة تحاكي متطلبات سوق العمل بكافة تخصصاته لتلبي متطلباته ، وقد تمثلت هذه الخطوة في مشروع إعداد المعايير المهنية الوطنية الذي يمثل الركيزة الأساسية في بناء البرامج التدريبية، إذ تعتمد المعايير في بنائها على تشكيل لجان تخصصية تمثل سوق العمل والمؤسسة العامة للتعليم الفني والتدريب المهني بحيث تتوافق الرؤية العلمية مع الواقع العملي الذي تفرضه متطلبات سوق العمل، لتخرج هذه اللجان في النهاية بنظرة متكاملة لبرنامج تدريبي أكثر التصاقاً بسوق العمل، وأكثر واقعية في تحقيق متطلباته الأساسية.

وتتناول هذه الحقيبة التدريبية " مقدمة قواعد بيانات أوراكل " لمتدربي قسم" البرمجيات " للكليات التقنية موضوعات حيوية تتناول كيفية اكتساب المهارات اللازمة لهذا التخصص.

والإدارة العامة لتصميم وتطوير المناهج وهي تضع بين يديك هذه الحقيبة التدريبية تأمل من الله عز وجل أن تسهم بشكل مباشر في تأصيل المهارات الضرورية اللازمة، بأسلوب مبسط يخلو من التعقيد، وبالاستعانة بالتطبيقات والأشكال التي تدعم عملية اكتساب هذه المهارات.

والله نسأل أن يوفق القائمين على إعدادها والمستفيدين منها لما يحبه ويرضاه؛ إنه سميع مجيب الدعاء.

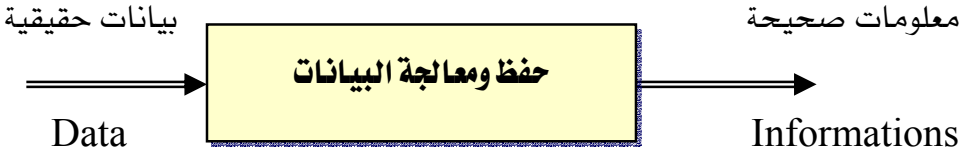
الإدارة العامة لتصميم وتطوير المناهج

تمهيد

أهمية قواعد البيانات:

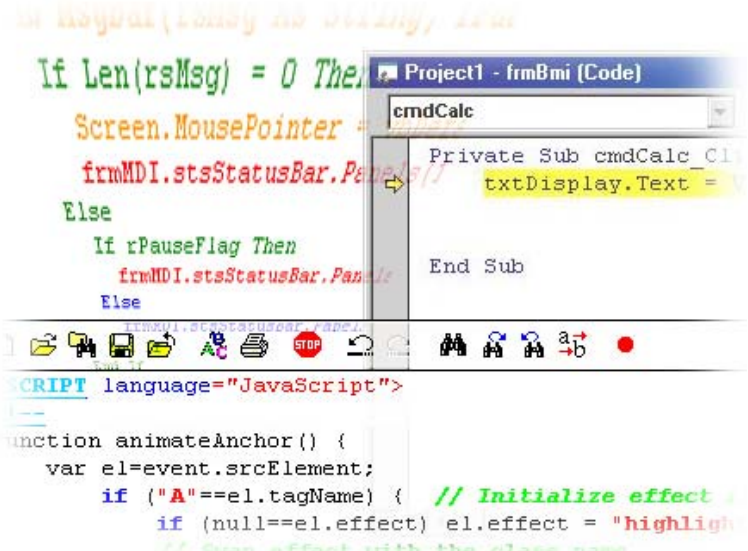
تعتبر عملية جمع البيانات ودقتها والتعامل معها من أهم العمليات التي يُعتمد عليها في معرفة معلومة معينة أو استنتاج واستنباط فكرة ما وهي ضرورية جداً لصاحب القرار في أي مجال لاتخاذ القرار المناسب في الوقت المناسب ، فمثلاً لو أن وزارة الصحة تريد معرفة عدد المرضى الموجودين في مدينة معينة في المملكة العربية السعودية وذلك لتحديد عدد الأطباء المفروض توفيرهم في كل مدينة مثلاً ، فذلك يتطلب أن يكون هناك بيانات دقيقة حول المرضى والسكان في كل مدينة لتزويد الوزارة بهذه المعلومة وعلى هذا الأساس تتخذ الوزارة التدابير والقرارات المناسبة لتوفير عدد الأطباء وباقي الاحتياجات الضرورية ، وبالمثل لو احتاجت الوزارة اتخاذ قرار ما حول محاربة مرض معين فلابد أن يتوفر لديها معلومات حول المدن التي يتنشر فيها المرض و نسبة انتشاره وعليه يتم اتخاذ القرار المناسب لمحاربة هذا المرض قبل انتشاره . فلو فرضنا أن هذه البيانات غير متوفرة أو أنها متوفرة بشكل غير منظم وغير دقيق فإن عملية اتخاذ القرار سوف تتآخر بشكل كبير ومن الممكن أن يُتخذ قرار غير مناسب مما يؤثر على عملية تسيير وتشغيل أمور المواطنين وما يترتب عليه من أضرار . ومن هذا نستنتج أن عملية جمع البيانات الدقيقة والتعامل معها بشكل صحيح من أهم العمليات التي تؤثر بشكل مباشر في الحياة اليومية وبخاصة ونحن في عصر المعلومات .

وفي الواقع إن البيانات الحقيقة الدقيقة تؤدي إلى معلومات صحيحة والبيانات غير الحقيقية تؤدي إلى معلومات غير صحيحة وعليه فلابد من دراسة وتحليل البيانات واكتشاف أنظمة لتسهيل هذه المهمة وضمان سلامة البيانات وسريتها لضمان الاستفادة القصوى من المعلومات . يوضح الشكل التالي العلاقة بين البيانات والمعلومات .



مقدمة قواعد بيانات أوراكل

مقدمة



الجدارة :

معرفة ما هي قواعد البيانات ومراحل تطورها وأنواعها .

الأهداف :

عندما يكتمل هذا الفصل يكون لديك القدرة على:

١ - فهم قواعد البيانات .

٢ - مراحل تطور قواعدالبيانات .

٣ - أنواع أنظمة إدارة قواعد البيانات .

٤ - قواعد البيانات العلائقية .

٥ - لغة التعامل مع قواعد البيانات (لغة الاستفسارات SQL) .

٦ - التعرف على بيئة SQL PLUS .

مستوى الأداء المطلوب :

أن يصل المتدرب إلى إتقان هذه الجدارة ١٠٠٪ .

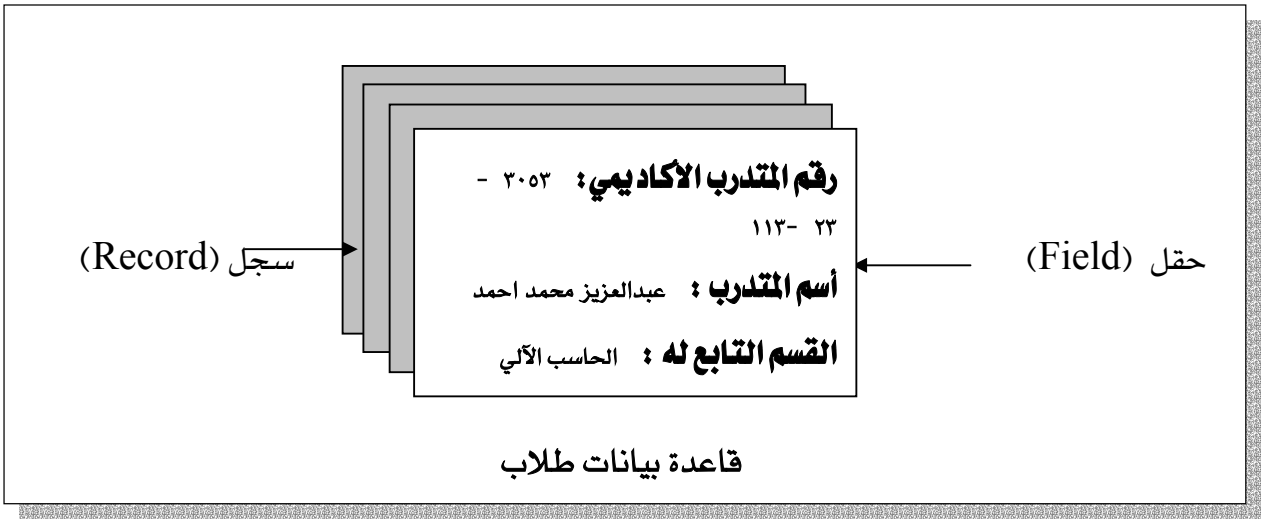
الوقت المتوقع للتدريب : ساعتان

الوسائل المساعدة :

- حاسب آلي .
- قلم .
- دفتر .

متطلبات الجدارة :

إلمام متوسط باللغة الإنجليزية ، التعامل الجيد مع أساسيات الحاسب الآلي .



الشكل رقم (١)

مفهوم قواعد البيانات Database Concepts

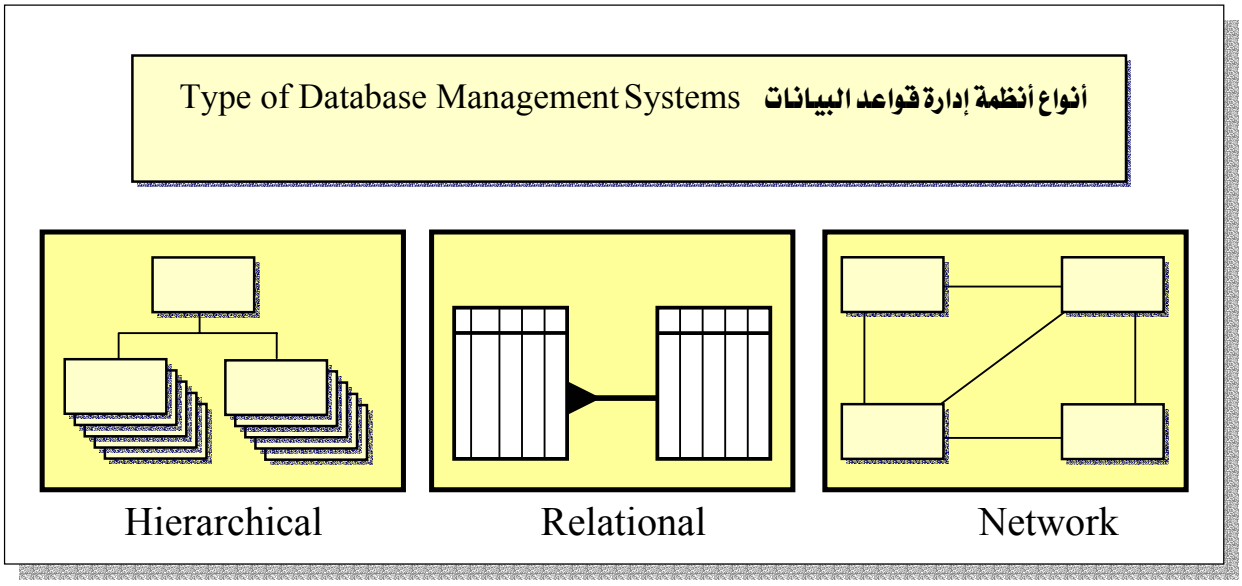
نفرض أننا نريد جمع البيانات عن المتدربين في كلية معينة، فإنه من المعروف أن لكل متدرب بيانات مثل (رقم المتدرب الأكاديمي، اسم المتدرب ، القسم الذي ينتمي إليه ، الشعبة ..... ألخ) ، فلو جمعنا بيانات كل متدرب في بطاقة وسميناها (سجل المتدرب RECORD) وكل بيان من بيانات المتدرب في هذا السجل سميناه (حقل FIELD) معنى ذلك أننا سوف نحصل على سجلات للمتدربين . عند جمع هذه السجلات مع بعضها نحصل على قاعدة بيانات للمتدربين تسمى DATABASE . كما في الشكل رقم (١) .

مراحل تطور قواعد البيانات :

لقد مرت عملية التعامل مع البيانات وكيفية تخزينها ومعالجتها بمراحل عديدة من قبل علماء قواعد البيانات فقد تم وضع نظريات وأساليب كثيرة للتعامل مع البيانات ومنها على سبيل المثال الآتي :

- حفظ البيانات في بطاقات نشر إليكترونية Electronic Spread sheets .
- تعتمد هذه الطريقة على حفظ البيانات داخل بطاقات إلكترونية يتم التعامل معها بشكل معين وتعتبر من أقدم الأساليب للتعامل مع البيانات .

- حفظ البيانات في ملفات تسمى مخازن معبأة Filling Cabinet .
- تعتمد هذه الطريقة على تخزين البيانات في ملفات ليتم التعامل معها ، وتعتبر أيضاً من الأساليب القديمة للتعامل مع قواعد البيانات .
- حفظ البيانات في قواعد بيانات Database وتعتبر هذه الطريقة هي الأحدث بالنسبة للطرق السابقة الذكر حيث تم عمل أنظمة للتعامل مع قواعد البيانات لتسهيل عملية تخزين البيانات واسترجاعها والتعديل فيها بسهولة ودقة (معالجتها) وتسمى هذه الأنظمة أنظمة إدارة قواعد البيانات Database Mangement System (DBMS) ومن هذه الأنظمة ما هو موضح الشكل رقم (٢) .



الشكل رقم (٢)

أنواع أنظمة إدارة قواعد البيانات :

- **نظام إدارة قواعدالبيانات الهرمية** Hierarchical database Mangement system

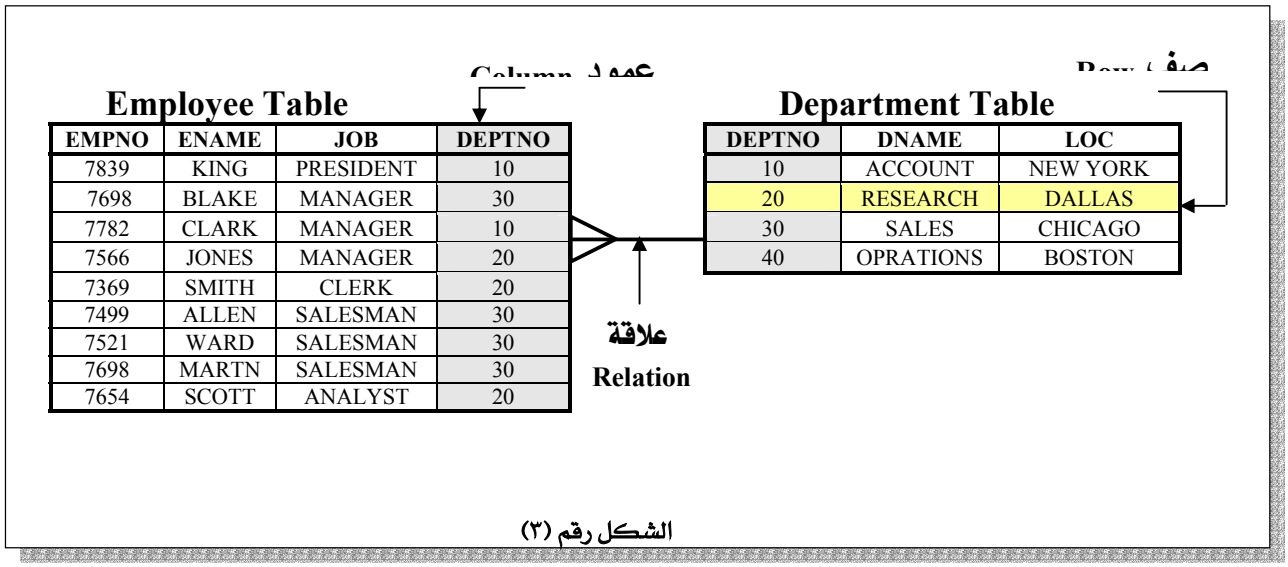
هـذا النظـام يسـتخدم في الماضـي وبخاصـة مـع أجهـزة الحاسـب الكـبيرة الـتي يطلـق عليهـا Main Frame حيث كان هذا النظام يتناسب معها بشكل جيد .

- **نظام إدارة قواعد البيانات الشبكية** Network database Mangement system

ظهر هذا النظام بعد النظام الهرمي وبخاصة بعد التوسع في أنظمة الشبكات ولكن كان هناك صعوبات كثيرة في عملية فهم وطبيعة التعامل مع البيانات كما في النظام الهرمي .

- **نظام إدارة قواعد البيانات العلائقية** Relational database Mangement system

يعتبر هذا النظام هو النظام الذي تعتمد عليه أغلب برامج قواعد البيانات مثل أوراكل لأنه من أقوى أنظمة قواعد البيانات لقدرته الفائقة على استيعاب كميات كبيرة من البيانات دون التأثير على أدائه من حيث السرعة والدقة ، ولأن هذا النظام يتمتع بالسرية والأمان لاحتوائه على نظام إعطاء الصلاحيات والحقوق لمستخدميه ولسهولته في الاستخدام والفهم وسهولة برمجة تطبيقاته .



الشكل رقم (٣)

قواعد البيانات العلائقية Relational Databse

تعتمد قواعد البيانات العلائقية على جمع البيانات في جداول بسيطة ثنائية الأبعاد يسهل فهمها تتكون من صفوف وأعمدة ، و كل عمود (Column) في الجداول عبارة عن حقل (Field) و كل صف (Row) من صفوف هذه الجداول عبارة عن سجل (Record) . وتم ربط هذه الجداول مع بعضها بروابط تسمى (Relations) ومن هنا جاءت تسميت قواعد البيانات العلائقية .

فقواعد البيانات العلائقية هي مجموعة من الجداول التي لها علاقة ما ببعضها . والشكل رقم (٣) يبين جدولين أحدهما يمثل بيانات الإدارات والآخر يمثل بيانات الموظفين ، كما يبين الشكل العلاقة بين الجدولين من خلال وجود العمود (DEPTNO) في كلا الجدولين .

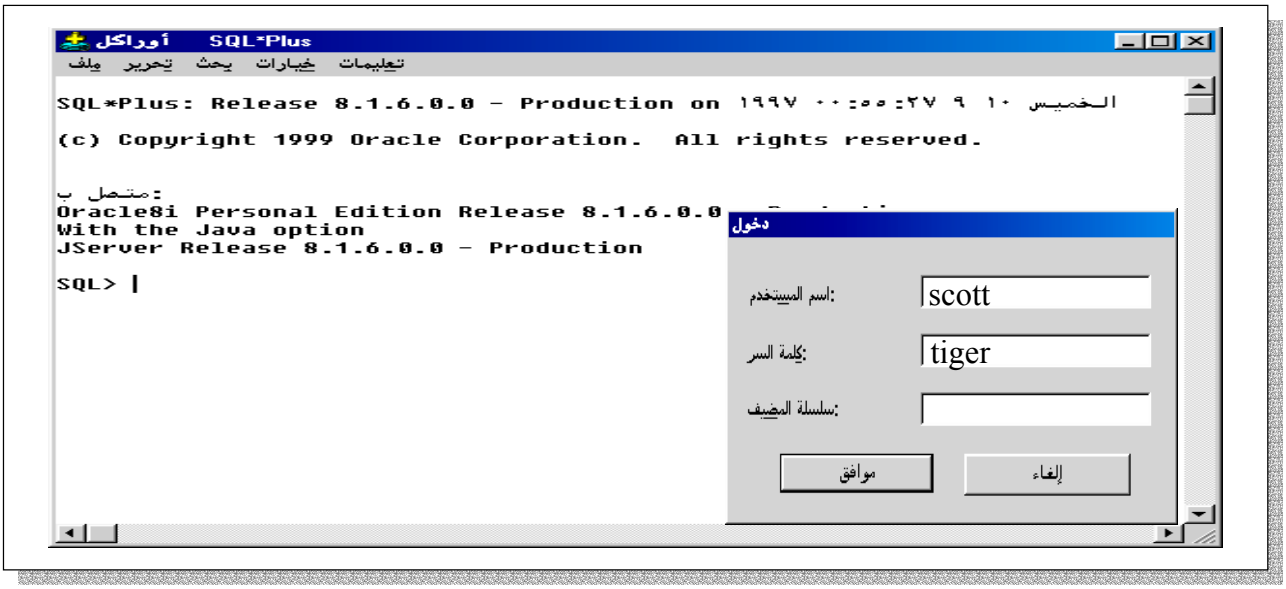
التعامل مع قواعد البيانات العلائقية Manipulate with relational database

قامت شركة أوراكل باعتماد لغة تسمى لغة الاستفسارات SQL (Structured Query Language) للتعامل مع قواعد البيانات العلائقة وهي لغة سهلة تقوم بإنشاء الأشياء (Objects) الخاصة بقاعدة البيانات مثل الجداول والتعامل معها وتقوم بعمل جميع الاستفسارات اللازمة والتي نريد أن نعرفها من قاعدة البيانات ويطلق عليها لغة (SQL) ، كما قامت شركة أوراكل أيضاً بعمل تطبيق أو بيئة تستقبل الأوامر الخاصة بلغة الاستفسارات SQL وهذه البيئة تسمى محرر الـ (SQL*PLUS) ويمكن من خلال هذا المحرر استقبال الأوامر الخاصة بلغة SQL وتنفيذها وتعديل الأخطاء الموجودة في الأمر وجميع العمليات الآخرى ، وسوف نقوم بشرح مفصل عن كل من لغة SQL والمحرر SQL*PLUS فيما يلي :

لغة الاستفسارات Structured Query Language (SQL)

هي لغة تستخدم لإصدار جميع الأوامر التي تتعلق بقاعدة البيانات ، وتنقسم هذه اللغة إلى خمسة أقسام رئيسة يمكن من خلالها إصدار الأوامر الخاصة بكل قسم ، والجدول التالي يوضح الأقسام المختلفة من هذه اللغة ووصف الأوامر لكل قسم .

القسم	الأمر	وصف الأمر
Data Retrieveal	SELECT	أمر استرجاع البيانات من جدول أو كائن
(DML) Data Manipulation Language	INSERT	أمر إضافة بيانات إلى جدول أو كائن
	UPDATE	أمر التعديل في بيانات جدول أو كائن
	DELETE	أمر حذف بيانات جدول أو كائن
(DDL) Data Definition Language	CREATE	أمر إنشاء جدول أو كائن
	Alter	أمر التعديل في جدول أو كائن
	DROP	أمر إلغاء جدول أو كائن
	RENAME	أمر تغيير الاسم جدول أو كائن
	TRUNCATE	إلغاء جزء أو بتر جزء من جدول أو كائن
Transaction Control	COMMIT	تثبيت البيانات في الجدول
	ROLLBACK	الرجوع عن تثبيت البيانات
	SAVEPOINT	الرجوع لنقطة معينة
(DCL) Data Control Language	GRANT	إعطاء الصلاحيات للمستخدمين للدخول على البيانات
	REVOKE	سحب الصلاحيات من المستخدمين



الشكل رقم (٤)

محرر (بيئة) ألـ SQL* PLUS .

الشكل رقم (٤) يبين شاشة الدخول على محرر sql*plus حيث تقوم بكتابة اسم المستخدم وهو (SCOTT) وكلمة المرور وهي (TIGER) ثم الضغط على مفتاح (موافق) وذلك للدخول على المحرر الذي يستقبل أوامر لغة الاستفسارات SQL . علماً بأن أسم المستخدم وكلمة المرور يمكن أن تتغير وذلك على حسب المستخدم هل له صلاحية الدخول أم لا ، فمن الممكن أن تدخل على المحرر باسم المستخدم (SYSTEM) وكلمة المرور (MANAGER) وفي هذه الحالة تدخل عليه وكأنك (مدير قاعدة البيانات) ، ويتكون محرر SQL*PLUS من قائمة تساعدك على تحرير الأمر والتعديل فيه وتنفيذه ، ووجود الرمز (SQL >) وهذا يشير إلى أنك تستطيع كتابة أي أمر بعده ، وهناك بعض الأوامر البسيطة التي تساعدك في كتابة وتعديل وتنفيذ الأمر ومنها على سبيل المثال .

- **SQL > EDIT**

يستخدم هذا الأمر لتعديل آخر أمر تم كتابته على محرر SQL*PLUS ، وعند تنفيذ هذا الأمر ستظهر لك شاشة (المفكرة) وبها آخر أمر تم كتابته حيث يمكن من خلال هذه الشاشة التعديل في الأمر ثم حفظه وتنفيذه مرة أخرى من خلال محرر SQL*PLUS ، ويمكن اختصار هذا الأمر فيكتب كالآتي : SQL > ED .

- **SQL > RUN**

يستخدم هذا الأمر لإعادة تنفيذ آخر أمر تم كتابته في محرر SQL*PLUS ، ويمكن كتابة هذا الأمر بالشكل التالي : SQL > R .

- **SQL > SPOOL Filename**

يستخدم هذا الأمر عندما نريد حفظ كل ما تم عمله داخل محررSQL*PLUS في ملف نصي بامتداد (LST) وذلك بغرض استرجاعها ومراجعتها ، ومن الممكن أن نحصل على نسخة مطبوعة بواسطة الأمر التالي : SQL > SPOOL OUT .

- **SQL > SAVE filename**

يستخدم هذا الأمر لحفظ الأوامر في ملف وذلك لاسترجاعها مرة آخرى وتنفيذها وهنا لابد من حفظ الملف بامتداد (sql) وذلك لنتمكن من تشغيله مرة آخرى . فإذا أردنا حفظ أمر ما داخل ملف اسمه test.sql نكتب الأمر التالي : SQL > SAVE test.sql .

- **SQL > GET filename**

يستخدم هذا الأمر لاسترجاع الأوامر التي تم حفظها بواسطة الأمر السابق . وذلك لتنفيذها مرة آخرى . فإذا أردنا استرجاع الأوامر من الملف test.sql نكتب الأمر التالي : SQL > GET test.sql .

- **SQL > START filename**

يستخدم هذا الأمر في تنفيذ الأوامر الموجودة في ملف تم حفظه بامتداد sql ، فإذا أردنا تنفيذ الأوامر الموجودة في الملف (test.sql) مثلاً نقوم بكتابة الأمر التالي :

SQL > START test.sql

- **SQL > @ filename**

هذا الأمر مثل الأمر السابق تماماً .

- **SQL > LIST**

يستخدم هذا الأمر في استعراض سطور آخر أمر تم كتابته ، ويمكن استعراض سطور معينة فمثلاً لو أردت استعراض السطور من ١ إلى ٣ نكتب الأمر كالتالي :

SQL > L 1 3

أسئلة الفصل الأول

١ – ماذا تعني الكلمات التالية (TABLE , ROW , COLUMN) .

٣ - لماذا يعتبر نظام إدارة قواعد البيانات العلائقة من أقوى أنظمة إدارة قواعد البيانات ؟

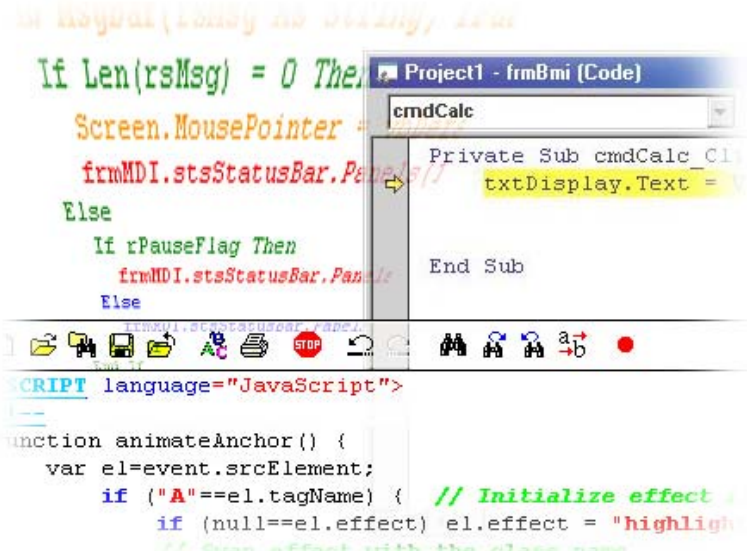
٤ – اذكر الفرق بين المحرر SQL*PLUS و لغة SQL ؟

٥ - ضح علامة صح (√) أمام الجُمل الصحيحة وعلامة خطأ (X) أمام الجمل الخاطئة ؟

- يعتبر أمر الاستعلام SELECT من أوامر محرر SQL*PLUS ويستخدم لاسترجاع البيانات. ( )
- تستخدم مجموعة أوامر DML لمعالجة بيانات الجداول . ( )
- يستخدم الأمر SPOOL SQL> لحفظ الأوامر التي نكتبها في المحرر داخل ملف ليسهل مراجعتها . ( )
- يستخدم الأمر L 2 4 لعرض السطور مبتدائ من السطر الثاني وحتى السطر الرابع ( )
- يستخدم الأمر START SQL> لتنفيذ عدة أوامر تم حفظها من قبل ( )
- يعتبر الأمر RUN من أوامر لغة الاستفسارات SQL ( )

مقدمة قواعد بيانات أوراكل

جملة الاستعلام الأساسية



جملة (SELECT) الأساسية

الجدارة :

معرفة جملة SELECT الأساسية وأجزائها المختلفة وكيفية التعامل معها .

الأهداف :

عندما يكتمل هذا الفصل يكون لديك القدرة على:

١ - فهم الصيغة العامة لجملة SELECT .
٢ - متطلبات وإرشادات كتابة جمل SELECT .
٣ - استرجاع البيانات من الجداول بواسطة جملة SELECT .
٤ - استرجاع الحقول باسماء مستعارة (Aliases) .
٥ - استخدام العمليات الحسابية وأولويات تنفيذها مع جملة SELECT .
٦ - استخدام أداة الربط بين الحقول (| |) Concatenation .
٧ - استخدام عبارة DISTINCT لمنع تكرار السجلات .
٨ - إظهار البناء الداخلي للجداول باستخدام الأمر (desc) describe .
٩ - التعامل مع القيمة NULL .

مستوى الأداء المطلوب :

أن يصل المتدرب إلى إتقان هذه الجدارة ١٠٠٪ .

الوقت المتوقع للتدريب : أربع ساعات

الوسائل المساعدة :

- حاسب آلي .
- قلم .
- دفتر .

متطلبات الجدارة :

إتقان ما سبقت دراسته في الفصل الأول .

الصيغة (الشكل) العامة لجملة SELECT .

```
SELECT      * or  Columns [alias]
FROM        Table
WHERE       condition or conditions
ORDER BY    Column or Alias   [ASC or DESC]  ;
```

تفسير الصيغة العامة :

SELECT	تستخدم في بداية الأمر لاسترجاع البيانات من الجداول .
*	هذا الرمز يستخدم عند استرجاع جميع الحقول من الجدول .
Columns	أسم الحقل أو الحقول المراد استرجاعها من الجدول .
Alises	الاسماء المستعارة للحقول .
FROM	تستخدم للإعلان عن اسم الجدول .
Table	أسم الجدول المراد استرجاع البيانات منه .
WHERE	تستخدم للإعلان عن الشرط أو الشروط .
Conditions	الشرط أوالشروط اللازمة لحصر البيانات الآتية من الجدول .
ORDER BY	تستخدم للإعلان عن كيفية ترتيب البيانات المسترجعة من الجدول .
Column or Alies	أسم الحقل أو الحقول أو الأسماء المستعارة المراد الترتيب بها .
;	فاصلة منقوطة للإعلان عن نهاية الأمر .

متطلبات وإرشادات كتابة جمل SQL .

يوجد هناك بعض الإرشادات التي يجب مراعاتها عند كتابة جملة SQL لتكون الجملة صحيحة وقابلة للتنفيذ ، وهذه الإرشادات هي :

١ - يمكن كتابة مكونات جملة SQL بالأحرف الكبيرة أو الصغيرة فهذا لا يؤثر على سلامة الجملة وذلك لأن جملة SQL غير حساسة للحروف Not Case Sensitive .

٢ - يفصل بين أسماء الحقول باستخدام الفاصلة (,) .

٣ - يمكن كتابة جملة SQL في عدة سطور فهذا لا يؤثر في صحة الجملة .

٤ - لا يمكن فصل الكلمات المحجوزة للغة أو اختصارها ، والكلمات المحجوزة تسمى Keywords وهي مثل (SELECT , FROM , WHERE , ORDER BY) .

٥ - يفضل كتابة الجملة على أسطر ليسهل قراءتها وفهمها .

٦ - لابد من الإعلان عن نهاية الجملة بواسطة (;) .

٧ - ملحوظة : أوامر محرر SQL*PLUS لا يوضع بعدها الفاصلة المنقوطة (;) .

لتنفيذ جملة SQL :

لتنفيذ جملة SQL من الممكن استخدام إحدى الطرق التالية :

١ - نضع الفاصلة المنقوطة (;) في نهاية الجملة .

٢ - نضع علامة (/) في نهاية الجملة عند مؤشر SQL > .

٣ - نكتب الأمر (RUN) عند مؤشر SQL > .

ولفهم طبيعة جملة SQL وكيفية تنفيذها ، فسوف نقوم بعرض أمثلة لحالات الجملة ونتائج تنفيذها لنصل من خلال الأمثلة إلى الفهم المطلوب :

قبل أن نبدأ في عرض الأمثلة هل تتذكر الجدولين المرسومين في الفصل السابق ، كان الجدول الأول عبارة عن جدول يحتوي على بيانات الموظفين ويسمى (EMP) ، والجدول الثاني كان يحتوي على بيانات الإدارات ويسمى (DEPT) (راجع صفحة ١ -٦ شكل (٣)) وهنا سوف نستخدم هذين الجدولين بشكل أساسي ولذلك لابد من مراجعتهما ومعرفة أسماء الحقول في كلا الجدولين .

مثال (١) : عرض جميع الحقول من جدول الإدارات DEPT .

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

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

في هذا المثال نقوم بعرض جميع الحقول والبيانات الموجودة في جدول الإدارات DEPT الذي يحتوي على الأعمدة التالية (DEPTNO , DNAME , LOC) وذلك باستخدام الرمز (*) والذي يعني إظهار جميع حقول الجدول ، لاحظ أن أسماء الحقول دائماً تظهر بالحروف الكبيرة .

مثال (٢) : عرض حقول معينة من جدول الإدارات DEPT .

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

DEPTNO	DNAME
10	ACCOUNTING
20	RESEARCH
30	SALES
40	OPERATIONS

يوضح المثال السابق كيفية إظهار عدة حقول معينة من الجدول ، ونلاحظ هنا بأن أسماء الحقول يفصل بينها الفاصلة (,) ، فالمثال يقوم بعرض جميع أرقام الإدارات وأسمائها فقط من جدول الإدارات (DEPT) .

استرجاع الحقول باسماء مستعارة (Aliases) :

نستخدم طريقة الاسماء المستعارة (Aliases) عندما نريد إظهار الحقل باسم غير اسمه الموجود في الجدول وذلك لتوضيح معنى الحقل مثلاً . وهناك ثلاث طرق لإظهار الحقول باسماء مستعارة :

١ - استخدام كلمة (AS) بين أسم الحقل والاسم المستعار .

٢ - استخدام المسافة (Space) بين أسم الحقل والاسم المستعار .

٣ - استخدام علامة التنصيص المزدوجة التالية (" ") عندما يكون الاسم المستعار أكثر من كلمة ، والمثال التالي يوضح ذلك .

مثال (٣) : عرض حقول باسماء مستعارة من جدول الموظفين .

```
SQL> SELECT ename AS name , sal salary ,  job "employee job"
  2  FROM    emp  ;
```

NAME	SALARY	employee job
SMITH	800	CLERK
ALLEN	1600	SALESMAN
WARD	1250	SALESMAN
JONES	2975	MANAGER
MARTIN	1250	SALESMAN
BLAKE	2850	MANAGER
CLARK	2450	MANAGER
SCOTT	3000	ANALYST
KING	5000	PRESIDENT
TURNER	1500	SALESMAN
ADAMS	1100	CLERK

يوضح المثال السابق كيفية استخدام الطرق المختلفة لإظهار الحقول باسماء مستعارة ، فنلاحظ هنا أن أسم الحقل ename قد ظهر في النتيجة باسم NAME بحروف كبيرة وكذلك حقل الراتب SALARY ، كما نلاحظ بأن الاسم المستعار الموجود بين علامتي التنصيص المزدوجة (" ") قد ظهر في النتيجة كما هو دون تحويله إلى الأحرف الكبيرة .

استخدام العمليات الحسابية وأولويات تنفيذها مع جملة SELECT :

من الممكن إجراء عمليات حسابية على الحقول العددية للحصول على معلومة معينة فمثلاً إذا أردنا إظهار الموظفين ورواتبهم في سنة فإننا نقوم بضرب راتب كل موظف في العدد ١٢ بشكل التالي (SAL*12) ، وكذلك عند إظهار إجمالي الراتب لكل موظف بعد إضافة ٥٠٠ ريال عليه (SAL + 500) . لاحظ أن العمليات الحسابية على الحقول لا تؤثر على البيانات المخزنة داخل الجدول .

المعاملات الحسابية التي تستخدم في العمليات الحسابية Arithmetic Operators

١ - الجمع (+) .
٢ - الطرح (-) .
٣ - الضرب (*) .
٤ - القسمة (/) .

يمكن استخدام المعاملات الحسابية في جميع أجزاء جملة SQL ماعدا الجزء الخاص بـ FROM والأمثلة التالية توضح كيفية إجراء العمليات الحسابية على الحقول .

مثال (٤) : عرض رواتب الموظفين السنوية من جدول الموظفين .

```
SQL> SELECT ename , sal , sal*12  "annual salary"
  2  FROM emp ;
  3
```

ENAME	SAL	annual salary
SMITH	800	9600
ALLEN	1600	19200
WARD	1250	15000
JONES	2975	35700

يبين المثال السابق كيفية استخدام العمليات الحسابية للحصول على رواتب الموظفين السنوية وذلك بضرب راتب كل موظف في ١٢ شهر .

أولويات تنفيذ العوامل الحسابية Operator Precedence

عند إجراء عملية حسابية كبيرة على حقل من الحقول لابد أن تعرف كيفية حسابها ولمعرفة ذلك لابد أن تعرف أولوية تنفيذ العوامل داخل جملة SQL فهي تنفذ بالترتب التالي :

١ - أولوية تنفيذ العمليات الحسابية للضرب والقسمة ثم للجمع والطرح .

٢ - العمليات من نفس الأولوية تنفذ من اليسار إلى اليمين .

٣ - عند وجود الأقواس في العمليات الحسابية يكون ما بداخلها له الأولوية وينفذ حسب الفقرة رقم (١) .

لاحظ الفرق بين العمليتين التاليتين :

1 – 100*(40+10) = 100*50=5000 .
2 – (100*40)+10 = 4000+10= 4010 .

والمثال التالي يوضح أولوية التنفيذ للعوامل الحسابية .

مثال (٥) : عرض رواتب الموظفين السنوية من جدول الموظفين .

```
SQL> SELECT  ename , sal , 12*sal+100
  2  FROM emp;                  ٢   ١

ENAME          SAL      12*SAL+100
---------- ---------- ----------
SMITH             800        9700
ALLEN            1600       19300
```

لاحظ أولوية التنفيذ :
العملية رقم (١) تفذ أولاً ثم العملية رقم (٢) .

مثال (٦) : عرض رواتب الموظفين السنوية من جدول الموظفين .

```
SQL> SELECT  ename , sal , 12*(sal+100)
  2  FROM emp;                  ٢   ١

ENAME          SAL      12*(SAL+100)
---------- ---------- ----------
SMITH             800        10800
ALLEN            1600        20400
```

لاحظ أولوية التنفيذ :
العملية رقم (١) تفذ أولاً لوجود الأقواس ثم العملية رقم (٢) .

بملحوظة الفرق بين المثاليين السابقين ، نجد أن النتيجة قد اختلفت تماماً وذلك لوجود الأقواس في المثال رقم (٦) فتم حساب ما بداخل الأقواس أولاً .

استخدام أداة الربط بين الحقول (||) Concatenation .

لعمل سلسلة من الحقول نقوم بربط حقلين أو أكثر باستخدام أداة الربط (||) والتي تسمى Concatenation ، ويكون ناتج الربط بين الحقول هو حقل واحد فقط ، ومن الممكن أن نربط مع الحقول نص معين نضعه بين علامتي تنصيص فردية (' ') ، والمثال التالي يوضح ذلك .

```
SQL> SELECT ename, job , ename||job as "employees"
  2  FROM emp ;

ENAME      JOB          employees
---------- ---------    -------------------
SMITH      CLERK        SMITHCLERK
ALLEN      SALESMAN     ALLENSALESMAN
WARD       SALESMAN     WARDSALESMAN
JONES      MANAGER      JONESMANAGER
MARTIN     SALESMAN     MARTINSALESMAN
```

في المثال السابق تم ربط حقلي الاسم والوظيفة بإدارة الربط (||) وقد ظهر هذان الحقلان كأنهما حقل واحد باسم مستعار employees يجمع بين الاسم والوظيفة .

```
SQL> SELECT ename, job , ename||' is a '||job as "employees"
  2  FROM emp ;

ENAME      JOB          employees
---------- ---------    -------------------
SMITH      CLERK        SMITH is a CLERK
ALLEN      SALESMAN     ALLEN is a SALESMAN
WARD       SALESMAN     WARD is a SALESMAN
JONES      MANAGER      JONES is a MANAGER
MARTIN     SALESMAN     MARTIN is a SALESMAN
```

في المثال السابق تم ربط الحقلين الاسم والوظيفة وبينهما نص هو (is a) باستخدام أداة الربط (||) ، فظهر الناتج كما هو واضح بالمثال .

استخدام عبارة DISTINCT لمنع تكرار السجلات :

عند إظهار محتويات الجدول نجد تكرار بعض القيم للحقل الواحد دون فائدة ، فمثلاً لو طلب منك معرفة أرقام الإدارات التي ينتمي إليها الموظفين في جدول الموظفون (EMP) ، فإنك بالطبع سوف تكتب هذا الأمر .

```
SQL> SELECT deptno
  2  FROM emp ;

 DEPTNO
------------
        20
        30
        30
        20
        30
        30
```

وبالنظر إلى النتيجة سوف تجد أن هناك تكراراً في الأرقام دون فائدة كما هو واضح ، ولمنع هذا التكرار نستخدم كلمة (distinct) مباشرةً بعد كلمة SELECT كماهو مبين في المثال التالي :

```
SQL> SELECT  DISTINCT deptno
  2  FROM emp ;

   DEPTNO
   ----------
         10
         20
         30
```

لاحظ أن أرقام الإدارات لم تتكرر وذلك لاستخدام كلمة DISTINCT .

إظهار البناء الداخلي للجداول باستخدام الأمر DESCRIBE (DESC) .

لإظهار معلومات حول أسماء الحقول وأنواعها الموجودة في جدول معين أي لإظهار البناء الداخلي للجدول نستخدم الأمر (DESCRIBE) والذي يمكن اختصاره إلى الأحرف التالية (DESC) .

```
SQL> DESC emp ;

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)
```

في المثال السابق تم إظهار أسماء الحقول الموجودة في جدول الموظفين وأنواعها وبعض المعلومات الخاصة بكل حقل مثل: هل نوع الحقل نصي أو تاريخ أو رقم ؟ وما هو طوله ؟

التعامل مع القيمة NULL .

ماذا تعني القيمة NULL ، هذه القيمة تسمى قيمة غير معروفة أو قيمة خاوية بمعنى أنها لاتساوي الصفر ولا مسافة ولا أي رقم أو نص ، فمثلاً هناك حقل في جدول الموظفين اسمه COMM هذا الحقل يخزن به قيمة تكليف الموظف بمهمة أو تكليفه بعمل إضافي يأخذ عليه أجر ، فهناك موظفون يأخذون بدل تكليف وموظفون آخرون لايمكن تكليفهم أي لايأخذون بدل تكليف إطلاقاً وفي هذه الحالة يتم ترك حقل التكليف COMM خالياً ليس به أي قيمة ونطلق على القيمة الخالية هذه NULL .

المثال التالي يبين الموظفين الذين لا يكلفون بعمل إضافي أي لا يأخذون بدل تكليف .

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

ENAME      JOB              SAL      COMM
---------- ---------- ---------- ----------
SMITH      CLERK            800
ALLEN      SALESMAN        1600        300
WARD       SALESMAN        1250        500
JONES      MANAGER         2975
MARTIN     SALESMAN        1250       1400
BLAKE      MANAGER         2850
CLARK      MANAGER         2450
SCOTT      ANALYST         3000
KING       PRESIDENT       5000
TURNER     SALESMAN        1500          0
```

NULL

يمكنك أن تلاحظ مثلاً الموظف SMITH لايأخذ بدل تكليف ولذلك تركت قيمة COMM خالية .

عند إجراء أي عملية حسابية عليها يكون النتائج دائماً (قيمة خالية NULL) ، يمكنك أن تلاحظ ذلك في المثال التالي :

```
SQL> SELECT ename, job , sal , 12*sal+comm
  2 FROM  emp ;

ENAME      JOB              SAL      12*SAL+COMM
---------- ---------- ---------- -----------
SMITH      CLERK            800
ALLEN      SALESMAN        1600       19500
```

NULL

أسئلة الفصل الثاني

١ – اكتب جملة استعلام لعرض البناء الداخلي لجدول الإدارات ، ثم اكتب جملة استعلام لعرض جميع بياناته ، بحيث تظهر النتيجة كالتالي :

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

٢ – اكتب جملة استعلام لعرض أسماء ووظائف وتواريخ تعيين وأرقام الموظفين بحيث يظهر رقم الموظف أولاً ؟

٣ – اكتب جملة استعلام لعرض وظائف الموظفين بدون تكرار ؟ بحيث تظهر النتيجة كالتالي :

JOB
ANALYST
CLERK
MANAGER
PRESIDENT
SALESMAN

٤ – اكتب جملة استعلام لعرض أرقام الموظفين وأسمائهم ووظائفهم مع تغيير أسماء الأعمدة كما هي في النتيجة التالية :

EMPLOYEE_NO	EMPLOYEE NAME	JOBS
7369	SMITH	CLERK
7499	ALLEN	SALESMAN
7521	WARD	SALESMAN
7566	JONES	MANAGER
7654	MARTIN	SALESMAN
7698	BLAKE	MANAGER

مقدمة قواعد بيانات أوراكل

حصر وترتيب البيانات

حصر وترتيب البيانات

RESTRICTING AND SORTING DATA

الجدارة :

استرجاع البيانات بشروط لها وترتيبها (حصر البيانات وترتيبها) .

الأهداف :

عندما يكتمل هذا الفصل يكون لديك القدرة على:

١ - فهم جملة الشرط (WHERE) لحصر البيانات .

٢ - معرفة واستخدام معاملات المقارنة Comparison Operators في جملة الشرط .

٣ - معرفة واستخدام معاملات المقارنة الخاصة (In,Between,Like,Is Null) .

٤ - معرفة واستخدام المعاملات المنطقية (AND,OR,NOT) .

٥ - استرجاع الصفوف بشكل مرتب تصاعدياً أو تنازلياً حسب عمود معين.

٦ - استرجاع الصفوف بشكل مرتب حسب عمودين أو أكثر.

مستوى الأداء المطلوب :

أن يصل المتدرب إلى إتقان هذه الجدارة ١٠٠٪ .

الوقت المتوقع للتدريب : أربع ساعات

الوسائل المساعدة :

- حاسب آلي .
- قلم .
- دفتر .

متطلبات الجدارة :

القدرة على التعامل مع جملة SELECT الأساسية ، معرفة الجداول المستخدمة EMP و DEPT وأسماء الحقول وخصائصها وأنواع البيانات داخلها .

الفصل الثالث : مقدمة

في الفصل السابق تم التعرف على جملة SELECT الأساسية والتي من خلالها يتم استرجاع البيانات من الجداول ، ومن الملاحظ أنه عند كتابة جملة SELECT في الفصل السابق كان الناتج دائماً يكون جميع الصفوف الموجودة بالجداول ، فلو أردنا استرجاع أسماء الموظفين الذين يعملون بوظيفة معينة وتكون هذه الاسماء مرتبة تصاعدياً أو تنازلياً حسب راتب كل منهم ، فماذا نعمل حتى نستطيع استرجاع الصفوف المطلوبة وترتيبها دون غيرها ؟ .

في هذا الفصل سوف نستخدم جملة الشرط (WHERE) لحصر الصفوف على أساس شرط معين، وأيضاً ترتيب الصفوف تصاعدياً أو تنازلياً باستخدام جملة (ORDER BY) .

جملة الشرط (WHERE) :

تُكتب هذه الجملة مباشرةً بعد جملة (FROM) وتستخدم في حصر البيانات على أساس شرط أو شروط معينة ، ويتكون الشرط من طرفين بينهم معامل مقارنة Comparison Operator ، وعند تحقق الشرط أي إن الشرط (TRUE) فإن جملة SELECT يكون لها ناتج ، أما إذا كان ناتج الشرط غير متحقق (FALSE) فإن جملة SELECT لا يكون لها أي ناتج وتظهر رسالة (No Row Selected) ومعناها لم يتم تحديد أي صف .

مكونات جملة الشرط WHERE :

يمكن أن تحتوي جملة الشرط (Where) على ما يلي :

- أسماء حقول Columns .
- معاملات مقارنة Comparison Operators .
- قيم ثابتة سواء كانت عددية أو نصية .
- تعبيرات حسابية .

متطلبات وإرشادات كتابة جملة الشرط WHERE .

يجب مراعاة الآتي عند كتابة جملة الشرط .

- عند استخدام قيم نصّية أو قيم تُعبر عن تاريخ لابد من وضعها داخل علامة التنصيص الفردية (' ') .
- في حالة استخدام القيم النصّية لابد من مراعاة حالة الأحرف كبيرة أم صغيرة .
- في حالة استخدام قيم تعبر عن تاريخ لابد من مراعاة صيغة التاريخ المستخدمة (FORMAT) علماً بأن الصيغة الأساسية للتاريخ داخل لغة SQL هي كالتالي : (DD-MON-YY) حيث إن (DD تعبر عن اليوم ، MON تعبر عن الشهر ، YY تعبر عن السنة) .

جملة الترتيب (ORDER BY) :

تستخدم هذه الجملة لترتيب الصفوف الناتجة ترتيباً تصاعدياً أو تنازلياً ، وتكتب دائماً في نهاية جملة SELECT .

متطلبات وإرشادات كتابة جملة الترتيب ORDER BY .

يجب مراعاة الآتي عند كتابة جملة الترتيب .

- يجب أن تُكتب في آخر جملة SELECT .
- تحتوي على أسماء حقول Columns أو أسماء مستعارة Alies .
- للترتيب تصاعدياً أكتب (ASC) وهي اختصار لكلمة (Ascending) وهي القيمة الافتراضية للترتيب (Default) .
- للترتيب تنازلياً لابد من كتابة (DESC) وهي اختصار لكلمة Descending .

مثال (١) : عرض أسماء و وظائف وأرقام إدارات الموظفين الذين يعملون بوظيفة (CLERK) ، مع ترتيب الناتج تصاعدياً حسب رقم الإدارة .

```
SQL> SELECT ename , job , deptno
  2 FROM     emp
  3 WHERE    job = 'CLERK'
  4 ORDER BY  deptno
```

لاحظ
كتابة النص بين علامتي ' ' وبالأحرف الكبيرة .

ENAME	JOB	DEPTNO
MILLER	CLERK	10
SMITH	CLERK	20
ADAMS	CLERK	20
JAMES	CLERK	30

المثال رقم (١) يبين طريقة استخدام جملة الشرط Where ، وذلك لعرض بيانات الموظفين الذين يعملون بوظيفة CLERK فقط ، لاحظ أن جملة الشرط قد احتوت على طرفين بينهما معامل حسابي وهو (=) ، وكانت النتيجة كما هي واضحة في المثال وذلك لأن جملة الشرط قد تحققت وكان هناك موظفون يعملون بوظيفة CLERK ، لاحظ أيضاً أن الطرف الثاني من جملة الشرط (**'CLERK'**) قد استخدمنا فيه ثابتاً نصياً و وضعنا هذا الثابت بين علامتي التنصيص الفردية . وأيضاً كتبنا الثابت النصي بالأحرف الكبيرة وذلك لأن البيانات داخل جدول الموظفين مسجلة بالأحرف الكبيرة ولذلك لابد من مراعاة حالة أحرف البيانات داخل الجدول كبيرة أم صغيرة حسب ما هو موجود في الجدول . كما تم ترتيب الناتج تصاعدياً بواسطة ORDER BY .

معاملات المقارنة المستخدمة في جملة الشرط Where . **Comparison Operators**

تستخدم معاملات المقارنة التالية للمقارنة بين طرفي الشرط في جملة Where .

المعامـــل	المعنـــى
=	يساوي
>	أكبر من
>=	أكبر من أو يساوي
<	أقل من
<=	أقل من أو يساوي
<> أو =!	لايساوي

الصيغة العامة لجملة الشرط WHERE .

```
SQL > WHERE   تعبير   OPERATOR   قيمة
```

أمثلة مختلفة :

- **WHERE** hiredate = '01-JAN-95' بشرط أن تاريخ التعيين يساوي (١ يناير ٩٥)
- **WHERE** sal >= 1500 بشرط أن الراتب أكبر من أو يساوي ١٥٠٠ دولار .
- **WHERE** ename = 'SMITH' بشرط أن أسم الموظف يكون SMITH

مثال (٢) : عرض أسماء و وظائف ورواتب الموظفين الذين رواتبهم أكبر من أو تساوي 3000 .

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

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

الناتج : أكبر من أو يساوي (٣٠٠٠)

مثال (٣) : عرض أسماء و رواتب وعمولة الموظفين الذين رواتبهم أقل من أو تساوي العمولة الخاصة بهم .

```
SQL> SELECT  ename , sal , comm
  2 FROM       emp
               3 WHERE     sal<= comm ;

ENAME          SAL       COMM
----------  ----------  ----------
MARTIN         1250        1400
```

في المثال السابق تم استخدام أسم العمود (comm) في طرف جملة الشرط الأيمن ، أي إننا من الممكن أن نستخدم أسماء الأعمدة للمقارنة مع أسماء أعمدة آخرى . وكانت النتيجة هي عرض بيانات الموظفين الذين رواتبهم أقل من أو تساوي العمولة الخاصة بهم .

معاملات مقارنة آخرى تستخدم في جملة الشرط WHERE .

هناك معاملات مقارنة آخرى تُستخدم في جملة الشرط هذه المعاملات تسهل عملية حصر البيانات بشكل أكبر ، و هي كالتالي :

المعامـــل	المعنـــي
قيمة AND قيمة BETWEEN	حصر البيانات بين رقمين
(مجموعة من القيم) IN	حصر البيانات ضمن مجموعة من القيم
{ % , _ } LIKE	حصر البيانات حسب مطابقة النص أو الحروف
IS NULL	حصر البيانات الخالية Null

مثال (٤) : عرض أسماء الموظفين ورواتبهم الذين تنحصر رواتبهم بين 1500 و 2500 .

```
SQL> SELECT  ename , sal
  2  FROM      emp
  3  WHERE   sal   BETWEEN 1500  AND  2500 ;
```

القيمة الصغرى — القيمة الكبرى

ENAME	SAL
ALLEN	1600
CLARK	2450
TURNER	1500

لاحظ : الراتب محصور بين ١٥٠٠ و ٢٥٠٠

في المثال السابق تم عرض بيانات الموظفين الذين تنحصر رواتبهم بين ١٥٠٠ و ٢٥٠٠ ، لاحظ أن القيمة ١٥٠٠ قد ظهرت في النتيجة لأن ناتج جملة BETWEEN يشتمل على القيمة الصغرى والقيمة الكبرى ، ونؤكد هنا على أن القيمة الصغرى لابد أن تكون أقل من القيمة الكبرى . فإذا عكسنا القيمة الكبرى مكان الصغرى (BETWEEN 2500 AND 1500) فإن جملة الشرط لن تتحقق أي تكون قيمتها (FALSE) وتظهر رسالة (No Row Selected) ومعناها لم يتم تحديد أي صف .

مثال (٥) : عرض رقم وأسم وراتب ورقم المدير للموظفين الذين لديهم مديرين بالأرقام التالية (٧٩٠٢ و ٧٥٦٦ و ٧٧٨٨) .

```
SQL> SELECT  empno , ename , sal , mgr
  2 FROM       emp
  3 WHERE      mgr  IN  (7902,7566,7788,7839)  ;
```

EMPNO	ENAME	SAL	MGR
7369	SMITH	800	7902
7788	SCOTT	3000	7566
7876	ADAMS	1100	7788
7902	FORD	3000	7566

في المثال السابق تم استرجاع بيانات الموظفين الذين لديهم مديرين من بين الأرقام التالية : (٧٩٠٢ و ٧٥٦٦ و ٧٧٨٨) أما المدير ذو الرقم (٧٨٣٩) فليس لديه موظفين لذلك لا يوجد له نتيجة . وبهذا فإن جملة IN تقوم بالبحث عن الرقم من بين قائمة الأرقام الموجودة داخل الأقواس .

معامل { _ , % } LIKE .

يستخدم هذا المعامل للبحث عن نص معين داخل ثابت أو حقل نصي ، حيث يتم مطابقة حروف النص المذكورة في جملة الشرط .

- **(%)** هذا الرمز يعني أي حرف أو أحرف ، مثلاً التعبير ('A%') يعني مطابقة النصوص التي تبدأ بحرف A مهما كانت باقي الحروف التالية له . فهو يستخدم للبحث عن نص يبدأ بالحرف A .

والتعبير ('%A') يعني مطابقة النصوص التي تنتهي بحرف A مهما كانت الحروف التي تسبقه .ويستخدم للبحث عن النصوص التي تنتهي بالحرف A .

أما التعبير ('%A%') فهو يستخدم للبحث عن النصوص التي تحتوي على الحرف A .

❖ (_) هذا الرمز يعني مطابقة حرف واحد فقط ، فمثلاً التعبير ('A%_') يعني أنه بغض النظر عن الحرف الأول، ويستخدم هذا التعبير عندما نريد البحث عن نص يكون الحرف الثاني فيه هو A .
أما التعبير ('A_ _') ، فيستخدم للبحث عن نص يكون الحرف الثالث فيه هو A .

ملحوظة :

يجب مراعاة حالة الأحرف هل هي كبيرة أم صغيرة عند استخدام المعامل LIKE .

مثال (٦) : عرض أسماء الموظفين الذين تبدأ أسمائهم بالحرف S .

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

ENAME
----------
SMITH
SCOTT
```

مثال (٧) : عرض أسم وتاريخ تعيين الموظفين الذين تم تعيينهم في العام ١٩٨١ م .

```
SQL> SELECT   ename , hiredate
 2  FROM      emp
 3  WHERE     hiredate LIKE '%81' ;

ENAME      HIREDATE
---------- ----------
ALLEN      20/02/81
WARD       22/02/81
JONES      02/04/81
MARTIN     28/09/81
BLAKE      01/05/81
CLARK      09/06/81
KING       17/11/81
TURNER     08/09/81
JAMES      03/12/81
FORD       03/12/81
```

مثال (٨) : عرض أسماء الموظفين الذين يكون الحرف الثاني في أسمائهم هو A .

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

ENAME
----------                    الحرف الثاني ( A )
WARD
MARTIN
JAMES
```

في المثال السابق تم البحث عن الاسماء التي يكون الحرف الثاني فيها هو (A) ثم عرض هذه الاسماء .

مثال (٩) : عرض أسم ورقم المدير للموظفين الذين لا يوجد لديهم مدير .

```
SQL> SELECT  ename , mgr
  2  FROM     emp
  3  WHERE   mgr  IS NULL  ;

ENAME          MGR
----------     ----------
KING                    ←——— NULL
```

في المثال رقم (٩) تم استخدام المعامل IS NULL والذي يقوم بحصر البيانات الخالية ، ففي المثال تم عرض بيانات الموظفين الذين لا يوجد لديهم مدير أي إن حقل المدير(MGR) لهذا الموظف خالي ليس به بيانات .

ملحوظة :

لا يمكن استخدام المعامل (=) مع القيم الخالية NULL ولكن لابد من استخدام المعامل IS NULL ، يمكن أن تجرب الأمر في المثال السابق بالشكل التالي لتعرف الفرق ؟

```
SQL> SELECT  ename , mgr          ←——— ( = ) الأمر هنا خطأ لاستخدام المعامل
  2  FROM     emp
  3  WHERE   mgr = NULL ;
```

المعاملات المنطقية في جملة الشرط WHERE .

المعامـــل	المعنـــــي
AND	ترجع النتيجة TRUE إذا كانت جملتا الشرط TRUE
OR	ترجع النتيجة TRUE إذا كانت إحدى جملتي الشرط TRUE
NOT	تنفي النتيجة ، أي ترجع النتيجة TRUE إذا كانت جملة الشرط FALSE

تستخدم المعاملات المنطقية التالية لتكوين أكثر من شرط في جملة WHERE ، وهي معاملات تربط بين جملتين شرطيتين أو أكثر وتكون النتيجة أما (TRUE) أي تحقق الشرط أو (FALSE) لم يتحقق الشرط .

المعامل AND :

هذا المعامل يربط بين جملتين شرطيتين ويكون الناتج TRUE إذا كانت كلتا الجملتين TRUE .
والجدول التالي يوضح ناتج المعامل AND مع الحالات المختلفة لجملتي الشرط .

جملة الشرط الأولى	جملة الشرط الثانية	ناتج المعامل AND
True	True	True
True	False	False
False	False	False
True	Null	Null
False	Null	False
Null	Null	Null

بالتدقيق في الجدول السابق يمكن أن نستخلص النتائج التالية :

أولاً : ناتج المعامل AND يكون دائماً FALSE إلا في حالة أن الجملتين TRUE فقط .

ثانياً : عند استخدام القيمة NULL مع المعامل AND يكون الناتج دائماً NULL إلا في حالة أن إحدى الجملتين تكون NULL والآخرى FALSE فقط .

مثال (١٠) : عرض رقم وأسم ووظيفة وراتب الموظفين الذين رواتبهم أكبر من أو تساوي 1100 وفي نفس الوقت وظيفتهم CLERK .

```
SQL> SELECT   empno , ename , job , sal
  2  FROM     emp
  3  WHERE    sal >=1100  AND  job='CLERK' ;
                  T                T
                 جملة             جملة
```

EMPNO	ENAME	JOB		SAL
7876	ADAMS	CLERK		1100
7934	MILLER	CLERK		1300

المثال السابق يبين أن جملتي الشرط **sal >=1100** و **job='CLERK'** قد تحققت أي إن نتيجة كلاً منهما كانت (TRUE) ولذلك فإن ناتج المعامل AND هو (TRUE) ولهذا قد تم عرض البيانات كما هو واضح من المثال .

مثال (١١) : عرض أسم وراتب وعمولة الموظفين الذين يزيد راتبهم عن 1100 وفي نفس الوقت تقل عمولتهم عن 500 .

```
SQL> SELECT ename , sal , comm
  2  FROM   emp
  3  WHERE  sal>1100  AND comm<500 ;
```

ENAME	SAL		COMM
ALLEN	1600		300
TURNER	1500		0

المعامل OR :

هـذا المعامـل يـربط بـين جملـتين شـرطيتين ويكـون النـاتج TRUE إذا كانـت إحـدى الجملـتين أو كلاهما TRUE . والجدول التالي يوضح ناتج المعامل OR مع الحالات المختلفة لجملتي الشرط .

جملة الشرط الأولى	جملة الشرط الثانية	ناتج المعامل OR
True	True	True
True	False	True
False	False	False
True	Null	True
False	Null	Null
Null	Null	Null

بالتدقيق في الجدول السابق يمكن أن نستخلص النتائج التالية :

أولاً : ناتج المعامل OR يكون دائماً TRUE إلا في حالة أن الجملتين FALSE فقط .

ثانياً : عند استخدام القيمة NULL مع المعامل OR يكون الناتج دائماً NULL إلا في حالة أن إحدى الجملتين تكون NULL والآخرى TRUE فقط فيكون الناتج TRUE .

مثال (١٢) : عرض رقم وأسم ووظيفة وراتب الموظفين الذين رواتبهم أكبر من (2500) أو تكون وظيفتهم MANAGER .

```
SQL> SELECT   empno , ename , job , sal
  2  FROM     emp
  3  WHERE    sal >2500  OR   job='MANAGER' ;
```

EMPNO	ENAME	JOB	SAL
7566	JONES	MANAGER	2975
7698	BLAKE	MANAGER	2850
7782	CLARK	MANAGER	2450
7788	SCOTT	ANALYST	3000
7839	KING	PRESIDENT	5000
7902	FORD	ANALYST	3000

المثال السابق يبين أنه لابد من تحقق أي من الجملتين حتى يتم استرجاع بيانات ، وبالتدقيق في الموظف رقم (7782) نجد أنه بالرغم من أن راتبه يقل عن ٢٥٠٠ إلا أنه ظهر في النتيجة وذلك لتحقق شرط الوظيفة (MANAGER) ، أي إنه يجب أن يتحقق أحد الشرطين لاسترجاع البيانات .

مثال (١٣) : عرض أسم وراتب ورقم الإدارة للموظفين الذين رواتبهم أقل من (1000) أو تكون إداراتهم رقم (10) .

```
SQL> SELECT   ename , sal , deptno
  2  FROM     emp
  3  WHERE  sal<1000  OR  deptno=10 ;
```

ENAME	SAL	DEPTNO
SMITH	800	20
CLARK	2450	10
KING	5000	10
JAMES	950	30
MILLER	1300	10

في المثال السابق تم عرض بيانات الموظفين الذين تقل رواتبهم عن ١٠٠٠ أو الموظفين المسجلين في الإدارة رقم (١٠) . نلاحظ من المثال أن الموظفين المسجلين في الإدارة رقم ٢٠ يأخذون راتباً أقل من ١٠٠٠ . والموظفين الذين يأخذون راتباً أكبر من ١٠٠٠ هم مسجلين في الإدارة رقم (١٠) وهذا يؤكد أنه لاسترجاع بيانات لابد من تحقق إحدى جملتي الشرط على الأقل .

المعامل NOT :

هذا المعامل يقوم بعكس ناتج جملة الشرط ، أي إنه إذا كانت جملة الشرط (TURE) فإن ناتج المعامل NOT يكون (FALSE) والعكس ، والجدول التالي يبين تأثير هذا المعامل على جملة الشرط .

جملة الشرط	ناتج المعامل NOT
True	False
False	True
Null	Null

يستخدم المعامل NOT أيضاً لنفي المعاملات الموضحة بالجدول التالي :

المعامـــل	نفي المعامل	المعني
BETWEEN ... AND ...	NOT BETWEEN .. AND ..	ليست بين رقمي .... و .....
IN (....)	NOT IN (....)	ليست ضمن القائمة .....
LIKE { % , _}	NOT LIKE { % , _}	ليست مطابقة
IS NULL	IS NOT NULL	ليست قيمة خالية

أمثلة على استخدام المعامل NOT لعكس المعاملات السابقة الذكر .

- WHERE job NOT IN ('CLERK' , 'MANAGER')
- WHERE sal NOT BETWEEN 1000 AND 1500
- WHREE ename NOT LIKE '%A%'
- WHERE comm IS NOT NULL

مثـال (١٤) : عـرض أسـم ووظيفـة المـوظفين الـذين ليسـت وظـائفهم مـن ضـمن الوظـائف التاليـة : (CLERK , MANAGER , ANALYST) .

```
SQL> SELECT   ename , job
  2  FROM     emp
  3  WHERE    job NOT IN ( 'CLERK' , 'MANAGER' , 'ANALYST' ) ;
```

ENAME	JOB
ALLEN	SALESMAN
WARD	SALESMAN
MARTIN	SALESMAN
KING	PRESIDENT
TURNER	SALESMAN

في هذا المثال تم نفي المعامل IN(...) ولذلك تم عرض بيانات الموظفين الذين لهم وظائف ليست من ضمن قائمة الوظائف التالية (CLERK , MANAGER , ANALYST) .

مثال (١٥) : عرض أسم ووظيفة وراتب الموظفين الذين لا تنحصر رواتبهم بين 1000 و 3000 .

```
SQL> SELECT    ename , job , sal
  2  FROM      emp
  3  WHERE     sal NOT BETWEEN 1000 AND 3000 ;
```

ENAME	JOB	SAL
SMITH	CLERK	800
KING	PRESIDENT	5000
JAMES	CLERK	950

في المثال السابق تم استبعاد الموظفين الذين تنحصر رواتبهم بين ١٠٠٠ و ٣٠٠٠ ، وتم عرض باقي الموظفين . وبذلك قد تم نفي المعامل (.... AND ... BETWEEN) .

مثال (١٦) : عرض أسم ووظيفة وراتب وعمولة الموظفين الذين يأخذون عمولة .

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

ENAME	JOB	SAL	COMM
ALLEN	SALESMAN	1600	300
WARD	SALESMAN	1250	500
MARTIN	SALESMAN	1250	1400
TURNER	SALESMAN	1500	0

في المثال السابق تم عرض بيانات الموظفين الذين يأخذون عمولة ، أي الذين ليست عمولتهم خالية NULL .

وإذا أردنا ترتيب الناتج تنازلياً حسب الراتب نضيف السطر التالي على المثال السابق .

ORDER BY sal DESC

أسئلة الفصل الثالث

١ – اكتب جملة استعلام لعرض أسماء ورواتب الموظفين الذين رواتبهم أكبر من 2850 والنتيجة مرتبة تنازلياً حسب الراتب ؟ بحيث تظهر النتيجة كالتالي :

ENAME	SAL
KING	5000
SCOTT	3000
FORD	3000
JONES	2975

٢ – اكتب جملة استعلام لعرض أسماء ورواتب الموظفين الذين رواتبهم لا تنحصر بين (1500,2850) ؟ بحيث تظهر النتيجة كالتالي :

ENAME	SAL
SMITH	800
WARD	1250
JONES	2975
MARTIN	1250
SCOTT	3000
KING	5000
ADAMS	1100
JAMES	950
FORD	3000
MILLER	1300

٣ – اكتب جملة استعلام لعرض أسماء وراوتب الموظفين الذين رواتبهم أكبر من 1500 ومسجلين في الإدارة رقم 10 أو مسجلين في الإدارة رقم 30 والنتيجة مرتبة تنازلياً حسب الراتب ؟ بحيث تظهر النتيجة كالتالي :

ENAME	SAL
KING	5000
BLAKE	2850
CLARK	2450
ALLEN	1600

٤ – اكتب جملة استعلام لعرض أسماء وتواريخ تعيين الموظفين المعينين سنة 1982 ؟ بحيث تظهر النتيجة كالتالي :

ENAME	HIREDATE
SCOTT	09-DEC-82
MILLER	23-JAN-82

٥ – اكتب جملة استعلام لعرض أسماء ورواتب وعمولة الموظفين الذين يأخذون عمولة ؟ بحيث تظهر النتيجة كالتالي :

ENAME	SAL	COMM
ALLEN	1600	300
TURNER	1500	0
MARTIN	1250	1400
WARD	1250	500

٦ – اكتب جملة استعلام لعرض أسماء الموظفين الذين يكون في أسمائهم الحرف الثالث A ؟ بحيث تظهر النتيجة كالتالي :

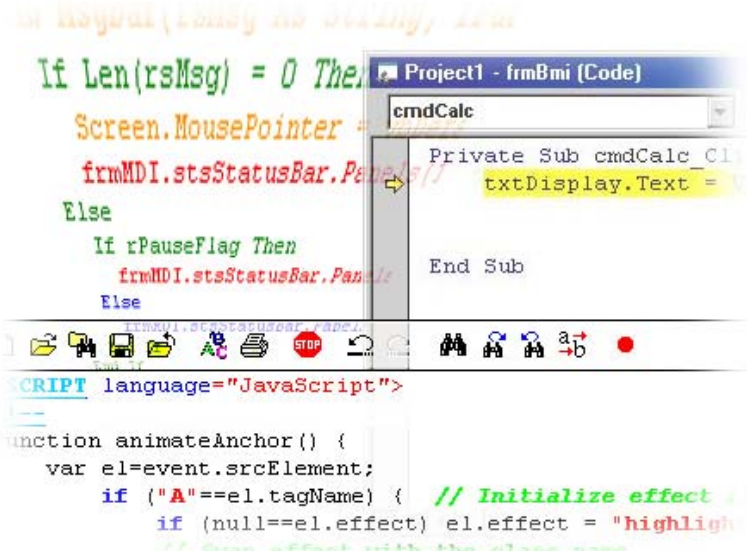
ENAME
BLAKE
CLARK
ADAMS

٧ – اكتب جملة استعلام لعرض أسماء الموظفين الذين تتضمن أسماؤهم الحرفين LL ؟ بحيث تظهر النتيجة كالتالي :

ENAME
ALLEN
MILLER

مقدمة قواعد بيانات أوراكل

الدوال التجميعية



الدوال التجميعية لأكثر من صف

GROUP FUNCTIONS

الجدارة :

استخدام الدوال التجميعية لأكثر من صف في جملة الاستعلام SELECT .

الأهداف :

عندما يكتمل هذا الفصل يكون لديك القدرة على:

١ - فهم ومعرفة الدوال التجميعية لأكثر من صف Group Functions .

٢ - معرفة أنواع الدوال التجميعية لأكثر من صف Group Functions .

٣ - إنشاء المجموعات من البيانات باستخدام الجزء GROUP BY .

٤ - فهم ومعرفة جملة الشرط المستخدمة مع الدوال التجميعية HAVING .

مستوى الأداء المطلوب :

أن يصل المتدرب إلى إتقان هذه الجدارة ١٠٠٪ .

الوقت المتوقع للتدريب : أربع ساعات

الوسائل المساعدة :

- حاسب آلي .
- قلم .
- دفتر .

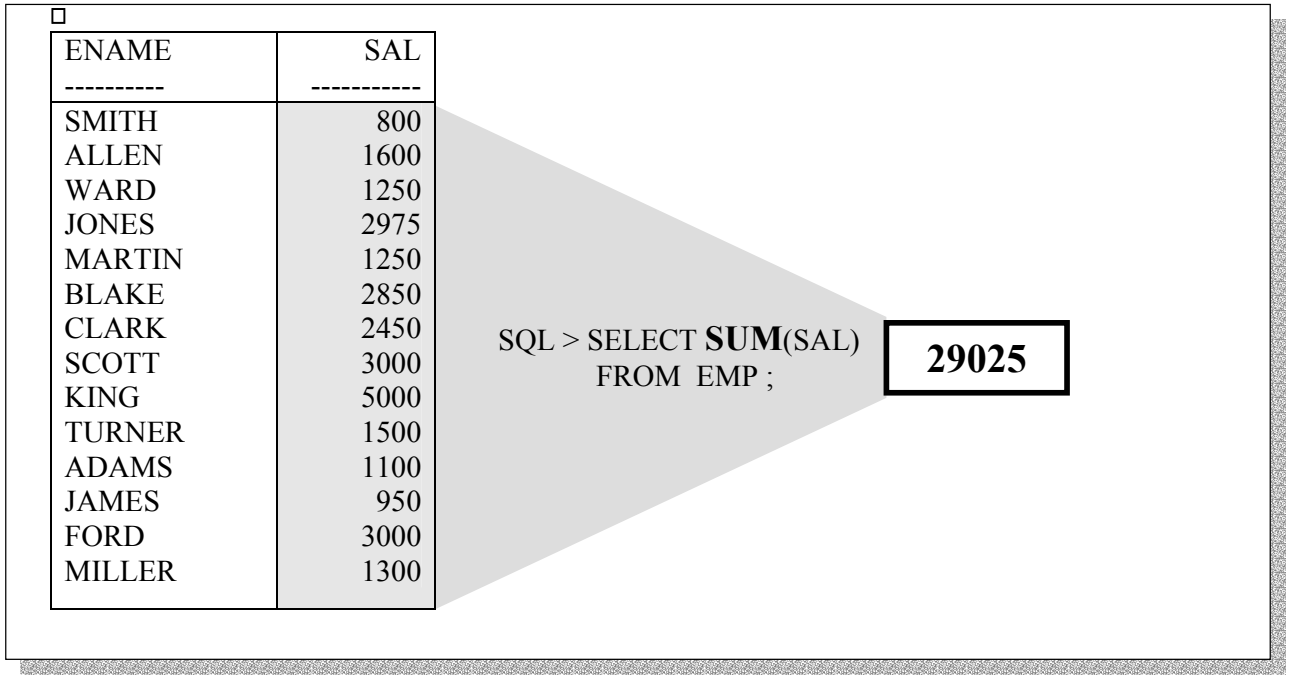
متطلبات الجدارة :

كل ما سبقت دراسته .

الفصل الخامس : مقدمة

الدوال التجميعية لأكثر من صف والتي تسمى (GROUP FUNCTIONS) هي دوال تتعامل مع بيانات مجموعة من الصفوف لآخراج قيمة واحدة فقط ، فمثلاً إذا أردنا إيجاد مجموع ما يأخذه الموظفون من رواتب فبدلاً من جمع كل الرواتب معاً ، نستخدم دالة من الدوال التجميعية تسمى SUM وهذه الدالة تقوم بعملية جمع للرواتب وتكون نتيجتها قيمة واحدة فقط وهي المجموع .

والشكل التالي يوضح كيفية عمل الدالة التجميعية SUM .



كما هو واضح من الشكل أن الدالة SUM قامت بجمع كل الرواتب لآخراج قيمة واحدة فقط وهي (29025) ، وبالمثل عندما نريد إيجاد أكبر راتب من بين رواتب الموظفين فهناك أيضاً دالة من الدوال التجميعية تقوم بعمل ذلك ، أي إن هذا النوع من الدوال يتعامل مع أكثر من صف لآخراج قيمة واحدة فقط .

أنواع الدوال التجميعية : Type of Group Functions

الجدول التالي يبين أنواع الدوال التجميعية ووظيفة كل دالة :

الدالـــــة FUNCTION	وظيفتهـــــا
SUM	دالة تستخدم لإيجاد المجموع لعدد من القيم
MAX	دالة تستخدم لإيجاد أكبر قيمة من بين مجموعة من القيم
MIN	دالة تستخدم لإيجاد أقل قيمة من بين مجموعة من القيم
AVG	دالة تستخدم لإيجاد المتوسط الحسابي لمجموعة من القيم
COUNT	دالة تستخدم لإيجاد عدد القيم أو عدد الصفوف وهذه الدالة تتجاهل القيم الفارغة NULL عند عملية العد .
STDDEV DEVIATION	دالة تستخدم لإيجاد الانحراف المعياري لمجموعة من القيم
VARIANCE	دالة تستخدم لإيجاد مقدار التباين (التشتت) لمجموعة من القيم

وجميع هذه الدوال تتجاهل القيم الفارغة NULL داخل الأعمدة .

أمثلة على الدوال التجميعية .

مثال (١) :

```
SQL> SELECT SUM(sal) , MAX(sal) , MIN(sal) , AVG(sal)
  2  FROM emp ;
```

SUM(SAL)	MAX(SAL)	MIN(SAL)	AVG(SAL)
29025	5000	800	2073.21429

في المثال السابق تم عرض مجموع رواتب الموظفين وأكبر راتب وأقل راتب والمتوسط الحسابي للرواتب .

ملحوظة هامة : كل من الدالتين MAX و MIN تتعامل مع جميع البيانات ، أي عند استخدامهما مع البيانات الحرفية تكون النتيجة حسب الترتب الأبجدي . لاحظ المثال التالي .

مثال (٢) :

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

MAX(ENAME)	MIN(ENAME)
WARD	ADAMS

في المثال السابق تم عرض أول أسماء الموظفين حسب الترتيب الأبجدي وآخر الاسماء .

مثال (٣) :

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

AVG(NVL(comm , 0))
157.14286

في المثال السابق تم عرض المتوسط الحسابي لمكافآت الموظفين ، ونلاحظ استخدام الدالة NVL هنا لحل مشكلة القيم الفارغة NULL الموجودة داخل العمود comm وذلك لأن المتوسط الحسابي عبارة عن مجموع القيم مقسومة على عدد الموظفين وبما أن الدوال التجميعية تتجاهل القيم الفارغة فكان لابد من استخدام الدالة NVL حتى يتم القسمة على عدد الموظفين وهم (14) موظفاً .

إذا لم نستخدم الدالة NVL فسوف يتم القسمة على أربعة (4) بدلاً من (14) وبالتالي يكون المتوسط الحسابي خطأ كما هو واضح كالتالي :

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

AVG(comm)
550

المتوسط الحسابي هنا خطأ لأنه تم القسمة على أربعة بينما عدد الموظفين هو ١٤ موظفاً .

التعامل مع الدالة COUNT .

يوجد حالتان للدالة count هما :

- COUNT(*) .
- COUNT(column) .

- تستخدم الدالة count(*) لعد جميع الصفوف داخل الجدول بما فيها الصفوف المكررة والصفوف التي تحتوي على القيم الفارغة NULL ، وإذا كانت جملة الاستفسار تحتوي على شرط where فإنها تقوم بعد الصفوف حسب جملة الشرط .
- تستخدم الدالة count(column) لعد قيم أو بيانات عمود معين مع تجاهل القيم NULL .

والأمثلة التالية توضح ذلك .

مثال (٤) :

```
SQL> SELECT COUNT(*),COUNT(comm), COUNT(deptno)
  2  FROM emp ;
```

COUNT(*)	COUNT(COMM)	COUNT(DEPTNO)
14	4	14

تم عرض إجمالي عدد الصفوف داخل جدول الموظفين وهم 14 صفاً ، وأيضاً تم عرض الموظفين الذين يأخذون مكافآت وعددهم داخل الجدول (4) ، نلاحظ هنا أن الدالة COUNT(comm) قد تجاهلت القيم الفارغة NULL داخل العمود COMM . كما تم عرض عدد الإدارات داخل العمود deptno وهم (14) بالرغم من أنها مكررة داخل العمود أي إن الدالة COUNT تقوم بعد القيم المكررة .

مثال (٥) :

```
SQL> SELECT COUNT(comm) , COUNT(*)
  2  FROM emp
  3   WHERE deptno=30 ;
```

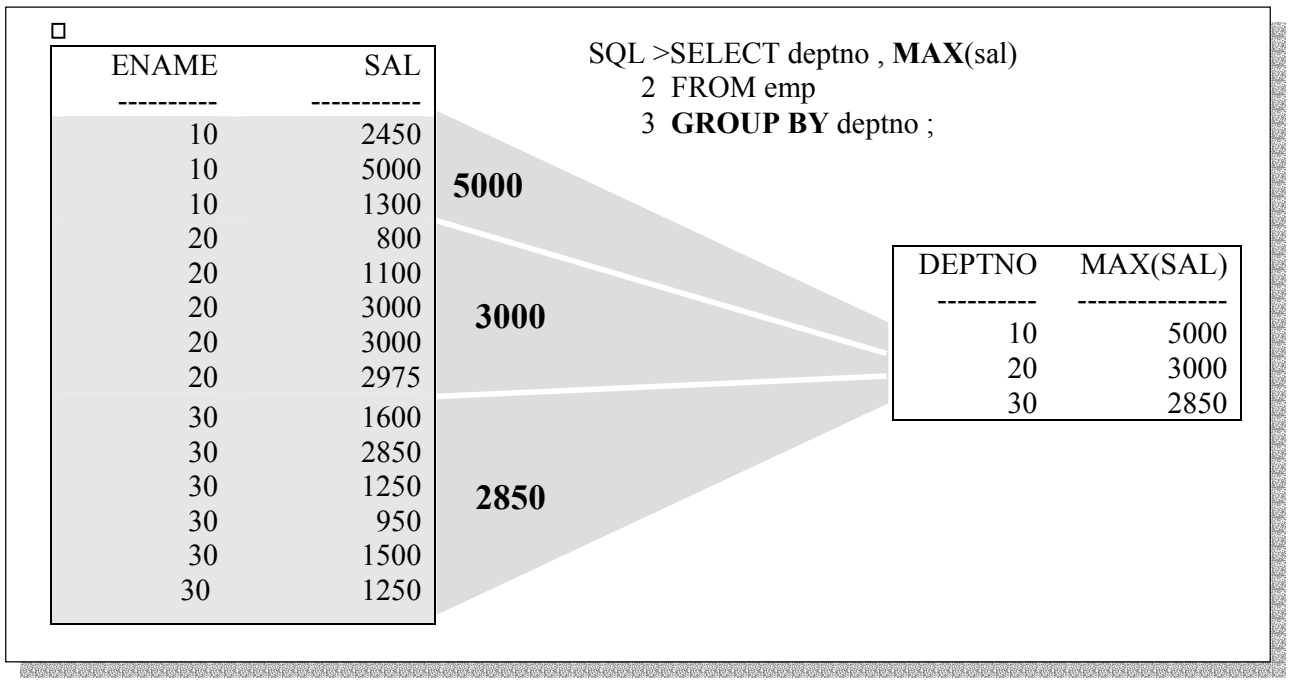
COUNT(COMM)	COUNT(*)
4	6

في المثال السابق تم عرض عدد الموظفين الذين يأخذون عمولة في الإدارة رقم 30 وكان عددهم أربعة بينما كان إجمالي عدد الموظفين في الإدارة هو ستة موظفين .

إنشاء مجموعات من البيانات باستخدام الجزء GROUP BY :

بفرض أننا نريد إيجاد أكبر راتب يأخذه موظف في كل إدارة معنى ذلك أننا نقسم الموظفين داخل الجدول إلى مجموعات حسب الإدارات ثم نقوم بإيجاد أكبر راتب في كل إدارة ، ولعمل ذلك نستخدم الجزء GROUP BY والذي يقوم بتقسيم البيانات إلى مجموعات على حسب عمود معين أو أكثر .

والشكل التالي يبين تأثير استخدام الجزء group by .



من الشكل السابق يتضح أنه تم تقسيم الموظفين إلى مجموعات حسب رقم الإدارة وكان أكبر راتب في الإدارة رقم 10 هو (5000) وأكبر راتب في الإدارة رقم 20 هو (3000) وأيضاً أكبر راتب في الإدارة رقم ٣٠ هو (2850) .

مثال (٦) :

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

DEPTNO	AVG(SAL)
30	1566.66667
20	2175
10	2916.66667

في المثال السابق تم عرض المتوسط الحسابي لمرتبات الموظفين في كل إدارة كما تم ترتيب المخرجات تصاعدياً حسب المتوسط الحسابي .

ملحظات على استخدام الدوال التجميعية :

- عند كتابة أي عمود داخل قائمة SELECT لابد من كتابته مع الجزء GROUP BY وذلك لأن الدوال التجميعية تتعامل مع عدة صفوف .
- يمكن استخدام الجزء ORDER BY لترتب الصفوف مع الدوال التجميعية كما هو مبين في المثال السابق .
- لا يمكن استخدام الدوال التجميعية في الجزء WHERE ولكن نستخدم الجزء HAVING بدلاً منها .

مثال (٧) :

عند تنفيذ هذا المثال تم إعطاء رسالة خطأ وذلك لأننا كتبنا أسم العمود (deptno) ضمن قائمة select ولم نكتبه ضمن الجزء group by ولتصحيح هذا الخطأ انظر المثال رقم (٦) .

```
SQL> SELECT deptno , AVG(sal)
  2  FROM emp
  3  ORDER BY AVG(sal)  ;

ERROR at line 1 :
ORA-00937: not a single-group  group function
```

رسالة خطأ

مثال (٨) :

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

ERROR at line 3 :
ORA-00934: group function  is not allowed here
```

رسالة خطأ

عند تنفيذ المثال السابق تظهر رسالة خطأ ، وذلك لأننا استخدمنا الدالة AVG(sal) داخل الجزء WHERE وهذا غير مسموح ، ولتصحيح هذا الخطأ لابد من استبدال جملة الشرط WHERE بجملة شرط خاصة بالدوال التجميعية وهي HAVING كما في المثال رقم (٩) :

مثال (٩) :

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

DEPTNO	AVG(SAL)
10	2916.66667
20	2175

في هذا المثال تم عرض المتوسط الحسابي لمرتبات الموظفين في كل إدارة بشرط أن تكون المتوسطات الحسابية للمرتبات أكبر من (2000) . لاحظ استخدام الجزء HAVING لتطبيق شرط معين على الدوال التجميعية .

يمكن استخدام جميع أجزاء جملة SELECT بشرط مراعاة الملاحظات السابقة عند الاستخدام . كما في المثال التالي .

مثال (١٠) :

```
SQL> SELECT job , SUM(sal)
  2  FROM emp
  3   WHERE job not like 'SALES%'
  4   GROUP BY job
  5   HAVING SUM(sal) >5000
  6   ORDER BY SUM(sal) ;
```

JOB	SUM(SAL)
ANALYST	6000
MANAGER	8275

في هذا المثال تم استخدام جميع أجزاء جملة SELECT لعرض مجموع رواتب الموظفين حسب كل وظيفة ، بشرط استبعاد الوظيفة التي تتضمن الحروف (SALES) وأيضاً استبعاد المجموع الأصغر من (5000) وترتيب المخرجات حسب مجموع الرواتب .

لاحظ عدم استخدام الدوال التجميعية في الجزء WHERE لوجود الجزء GROUP BY .

أسئلة الفصل الخامس

١ – اكتب جملة استعلام لعرض أعلى وأقل قيمة للرواتب وأيضاً المتوسط الحسابي للرواتب وقم بملاحظه الناتج ؟ بحيث تظهر النتيجة كالتالي :

MAXIMUM	MINIMUM	SUM	AVERAGE
5000	800	29025	2073

٢ – اكتب جملة استعلام لعرض أسماء الوظائف وأعلى وأقل راتب لهذه الوظائف كل على حده ؟ بحيث تظهر النتيجة كالتالي :

JOB	MAXIMUM	MINIMUM
ANALYST	3000	3000
CLERK	1300	800
MANAGER	2975	2450
PRESIDENT	5000	5000
SALESMAN	1600	1250

٣ – اكتب جملة استعلام لعرض الوظائف وعدد الموظفين في كل وظيفة ؟ بحيث تظهر النتيجة كالتالي :

JOB	COUNT(*)
ANALYST	2
CLERK	4
MANAGER	3
PRESIDENT	1
SALESMAN	4

٤ – اكتب جملة استعلام لعرض عدد المديرين ؟ بحيث تظهر النتيجة كالتالي :

NUMBER OF MANAGERS
6

مقدمة قواعد بيانات أوراكل

عرض البيانات من أكثر من جدول

عرض البيانات من أكثر من جدول

DISPLAYING DATA FROM MULTIPLE TABLES

الجدارة :

كيفية عرض البيانات واسترجاعها من أكثر من جدول عن طريق الطرق المختلفة لربط الجداول مع بعضها .

الأهداف :

عندما يكتمل هذا الفصل يكون لديك القدرة على:

١ - فهم كيفية عرض البيانات من أكثر من جدول ومفهوم ربط الجداول .
٢ - معرفة الأنواع المختلفة لربط جدولين أو أكثر .
٣ - معرفة واستخدام الربط بالتساوي بين الجداول Equijoin .
٤ - معرفة واستخدام الربط بعدم التساوي بين الجداول Non-Equijoin .
٥ - معرفة واستخدام الربط الخارجي بين الجداول Outer Join .
٦ - معرفة واستخدام الربط الداخلي لنفس الجدول Self Join .
٧ - معرفة واستخدام الاسماء المستعارة للجداول عند الربط بين جدولين أو أكثر .

مستوى الأداء المطلوب :

أن يصل المتدرب إلى إتقان هذه الجدارة ١٠٠٪ .

الوقت المتوقع للتدريب : أربع ساعات

الوسائل المساعدة :

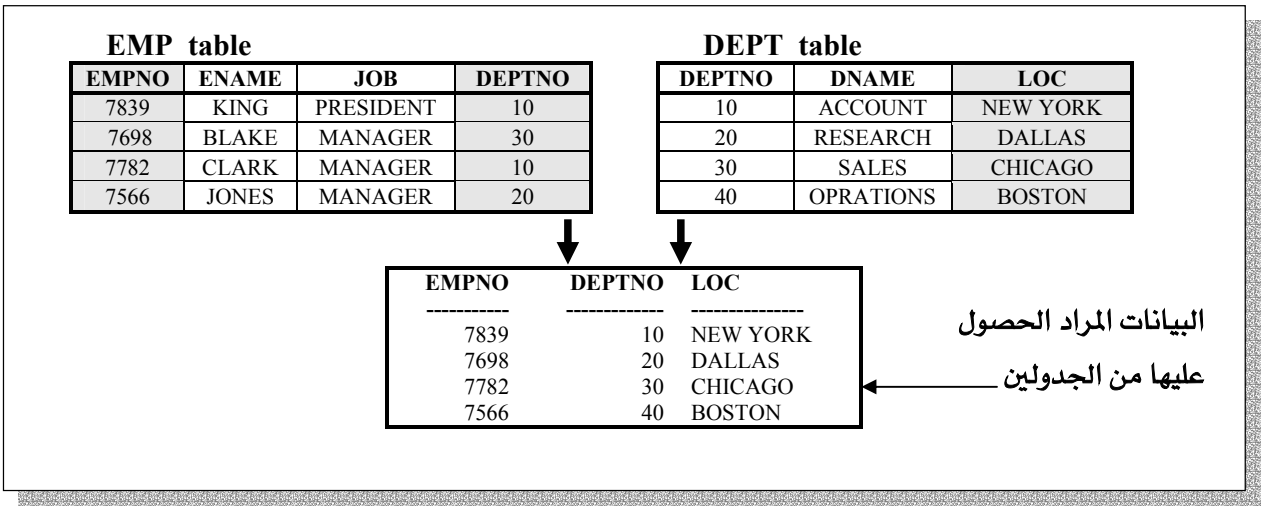
- حاسب آلي .
- قلم .
- دفتر .

متطلبات الجدارة :

كل ما سبقت دراسته .

الفصل السادس : مقدمة

في بعض الأحيان نريد أن نقوم بعرض بيانات من أكثر من جدول لعمل تقارير مفيدة وشاملة ، فمثلاً لو أردنا عرض رقم الموظف ورقم الإدارة التابع لها وموقع هذه الإدارة نجد أننا لابد من الحصول على هذه البيانات من جدول الموظفين وجدول الإدارات لكون رقم الموظف موجود في جدول الموظفين ورقم الإدارة موجود في جدول الإدارات وأيضاً موجود في جدول الموظفين بينما موقع الإدارة موجود في جدول الإدارات ، كما هو موضح بالشكل التالي :



وللحصول على تلك البيانات لابد من عمل ربط بين الجدولين . وسوف نقوم في هذا الفصل بشرح أنواع الربط المختلفة وكيفية عمل كل نوع من هذه الأنواع .

تعريف الربط : Join defination

هو عبارة عن ربط بين جدولين أو أكثر للحصول على بيانات من تلك الجداول .

أنواع الربط : Types of Joins

توجد عدة أنواع من الربط (Joins) وهي كالتالي :

- الربط بالتساوي Equijoin .
- الربط بعدم التساوي Non-Equijoin .
- الربط الخارجي Outer Join .
- الربط الداخلي في نفس الجدول Self Join .

ويتم عمل هذه الأنواع عن طريق جملة الاستفسار SELECT وبخاصة في جزء الشرط WHERE .

الربط بالتساوي : Equijoin

في هذا النوع من الربط يتم ربط جدولين أو أكثر عن طريق عمودين متساويين ، العمود الأول عادةً ما يكون مفتاح أساس (Primary Key) في الجدول الأول والعمود الثاني يكون عبارة عن عمود ربط (Foreign Key) في الجدول الثاني .

والشكل التالي يبين الربط بالتساوي بين جدول الموظفين وجدول الإدارات عن طريق العمود (deptno) الموجود في كل منهم .

EMP table

EMPNO	ENAME	JOB	DEPTNO
7839	KING	PRESIDENT	10
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7566	JONES	MANAGER	20

DEPT table

DEPTNO	DNAME	LOC
10	ACCOUNT	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPRATIONS	BOSTON

Foreign Key **Primary Key**

سوف يتم عمل الربط بين الجدولين باستخدام جملة SELECT عن طريق العمودين المشار إليهما بالسهم ، كما في المثال التالي .

مثال (١) :

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

شرط الربط بين الجدولين

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

في المثال السابق تم عرض بيانات من جدولين عن طريق الربط بالتساوي ، وسوف نقوم بشرح كل جزء من أجزاء جملة SELECT على حدا كالتالي:

- في قائمة SELECT تم عرض رقم الموظفين وأسمائهم وأرقام إداراتهم من جدول الموظفين (EMP) ، كما تم عرض رقم الإدارات وموقعها من جدول الإدارات (DEPT) ، ونلاحظ هنا أننا حددنا من أين تأتي البيانات عن طريق كتابة أسم الجدول قبل أسم العمود ويفصل بينهما العلامة (.) كالتالي (emp . empno) .
- في الجزء FROM تم كتابة أسماء الجداول التي ستأتي منها البيانات كالتالي (FROM emp,dept) .
- في الجزء WHERE تم كتابة شرط الربط بين الجدولين وهذا الشرط مهم جداً لإتمام عملية الربط ، وبدون هذا الشرط سوف تكون النتيجة ليس لها معنى أو فائدة .

أستخدام الاسماء المستعارة للجداول :

يمكن استخدام الاسماء المستعارة للجداول لتسهل عملية كتابة الأعمدة ، فمثلاً نقوم باستبدال أسم الجدول (emp) بالحرف (e) ، وأسم الجدول (dept) بالحرف (d) كالتالي :

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

وطبعاً سوف تكون النتيجة مماثلة تماماً للمثال رقم (١) .

عندما نريد عرض البيانات الموجودة في المثال رقم (١) ولكن للموظف KING فقط فإننا هنا لابد من زيادة شرط على جملة SELECT كالتالي :

مثال (٢) :

```
SQL> SELECT e.empno , e.ename , e.deptno ,
  2         d.deptno , d.loc
  3  FROM emp  e , dept  d <----------  (e , d) الاسماء المستعارة
  4  WHERE e.deptno=d.deptno
  5  AND e.ename = upper('king') ;
```

EMPNO	ENAME	DEPTNO	DEPTNO	LOC
7839	KING	10	10	NEW YORK

الربط بعدم التساوي : Non-Equijoin

يتم استخدام هذا النوع من الربط عندما لا توجد علاقة مباشرة بين الجدولين المراد ربطهما أي إننا لا نستخدم فيه علامة التساوي (=) ، ولكن لابد من وجود علاقة غير مباشرة مثل شرط معين ينطبق عليهما ، مثلاً عندنا جدول الموظفين وفيه عمود الراتب (SAL) وأيضاً لدينا جدول آخر يسمى جدول الفئات (SALGRADE) في هذا الجدول يتم وضع فئات للرواتب وكل فئة تنحصر بين أعلى راتب وأقل راتب ، فمثلاً الموظف الذي يأخذ راتب (3000) يتبع الفئة رقم (٤) كما هو واضح من جدول الفئات أدناه وبذلك نجد أن هناك علاقة بين الجدولين وهي أن كل راتب في جدول الموظفين لابد أن يتبع فئة معينة داخل الجدول (SALGRADE) أي إنه يقع بين أعلى قيمة وأقل قيمة داخل الجدول .

والشكل التالي يوضح جدول الموظفين EMP وجدول الفئات SALGRADE والعلاقة بينهم .

EMP table

EMPNO	ENAME	JOB	SAL
7839	KING	PRESIDENT	5000
7698	BLAKE	MANAGER	2850
7782	CLARK	MANAGER	2450
7566	JONES	MANAGER	2975
7654	MARTIN	SALESMAN	1250

SALGRADE table

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

SAL	GRADE
5000	5
2850	4
2450	4
2975	4
1250	2

البيانات التي توضح فئة كل راتب من رواتب الموظفين والتي حصلنا عليها من الجدولين

المثال التالي يوضح كيفية الربط بين الجدولين عن طريق الربط بعدم التساوي Non-Equijoin .

مثال (٣) :

```
SQL> SELECT e.ename , e.sal , s.grade
  2  FROM emp e , salgrade s
  3  WHERE e.sal BETWEEN s.losal AND s.hisal ;
```

شرط الربط بين الجدولين

ENAME	SAL	GRADE
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

في المثال السابق تم عرض أسماء الموظفين ورواتبهم من جدول الموظفين وعرض الفئات لكل راتب من جدول الفئات وذلك عن طريق الربط بعدم التساوي بين الجدولين وتم ذلك في جزء الشرط WHERE e.sal BETWEEN s.losal AND s.hisal والذي يوضح شرط انحصار رواتب الموظفين في جدول الموظفين بين أقل راتب وأكبر راتب في جدول الفئات .

الربط الخارجي : Outer Join

يتم استخدام هذا النوع من الربط عندما توجد بيانات في أحد الجداول ولكنها لا تظهر في حالة الربط بالتساوي (Equijoin) بين الجدولين أي إنها غير مطابقة لشرط التساوي ونريد لهذه البيانات أن تظهر ، في هذه الحالة نقوم بالربط بين الجدولين باستخدام الربط بالتساوي ولكن نضيف الجزء (+) بجانب العمود الفاقد للبيانات ويسمى الربط في هذه الحالة بالربط الخارجي (Outer Join) ، فمثلاً توجد الإدارة رقم (40) في جدول الإدارات ولكن لا يوجد بها موظفين مسجلين في جدول الوظائف عند استخدام الربط بالتساوي فإن هذه الإدارة لا تظهر في المخرجات لعدم تطابق شرط التساوي عليها ، ولإظهارها لابد من استخدام الربط الخارجي .

المثال التالي يبين كيفية الربط بين جدولين باستخدام الربط الخارجي (Outer Join) لإظهار كافة البيانات الموجودة بالجدولين سواء كانت البيانات المطابقة لشرط التساوي أو غير المطابقة لشرط التساوي .

مثال (٤) :

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

علامة الربط الخارجي

EMPNO	ENAME	DEPTNO	DNAME
7782	CLARK	10	ACCOUNTING
7839	KING	10	ACCOUNTING
7934	MILLER	10	ACCOUNTING
7369	SMITH	20	RESEARCH
7876	ADAMS	20	RESEARCH
7902	FORD	20	RESEARCH
7788	SCOTT	20	RESEARCH
7566	JONES	20	RESEARCH
7499	ALLEN	30	SALES
7698	BLAKE	30	SALES
7654	MARTIN	30	SALES
7900	JAMES	30	SALES
7844	TURNER	30	SALES
7521	WARD	30	SALES
		40	OPERATIONS

هذه الإدارة ظهرت لاستخدامنا الربط الخارجي

في المثال السابق تم عرض أرقام الموظفين وأسمائهم من جدول الموظفين كما تم عرض أرقام الإدارات وأسمائها من جدول الإدارات باستخدام الربط الخارجي (Outer Join) ولذلك قد ظهرت الإدارة رقم (40) بالرغم أنها غير مطابقة لشرط التساوي أي لا يوجد بها موظفين مسجلين في جدول الموظفين .

الربط الداخلي لنفس الجدول : Self Join

EMP table

EMPNO	ENAME	JOB	MGR
7839	KING	PRESIDENT	
7698	BLAKE	MANAGER	7839
7782	CLARK	MANAGER	7839
7566	JONES	MANAGER	7839
7654	MARTIN	SALESMAN	7698

EMP (WORKER)

EMPNO	ENAME	MGR
7839	KING	
7698	BLAKE	7839
7782	CLARK	7839
7566	JONES	7839
7654	MARTIN	7698

EMP (MANAGER)

EMPNO	ENAME
7839	KING
7698	BLAKE
7782	CLARK
7566	JONES
7654	MARTIN

عندما ندقق في جدول الموظفين نجد أنه يحتوي على عمود يسمى (MGR) هذا العمود يمثل رقم المدير للموظف ، فنجد أن الموظف (BLAKE) مديره هو الموظف ذو الرقم (7839) أي إنه الموظف (KING) ، ومن ذلك يتضح لنا أن هناك علاقة بين عمود المدير (MGR) ورقم الموظف (EMPNO) فالمدير هو نفسه عبارة عن موظف أي يوجد له رقم موظف داخل العمود (EMPNO) ، أي إننا نستطيع ربط الجدول بنفسه عن طريق العمودين (MGR) و (EMPNO) .

ولعمل هذا النوع من الربط لابد من تقسيم جدول الموظفين إلى جدولين أحدهما يمثل جدول الموظفين ونسميه مثلاً (WORKER) والآخر يمثل جدول المدراء ونسميه مثلاً (MANAGER) كما هو واضح من الشكل السابق ، ونقوم بعد ذلك بربط الجدولين عن طريق الربط بالتساوي (Equijoin) .

والمثال التالي يوضح كيفية عمل الربط الداخلي لنفس الجدول .

مثال (٥):

```
SQL> SELECT WORKER.empno , WORKER.ename , MANAGER.ename manager
  2  FROM emp worker , emp manager
  3  WHERE worker.mgr = manager.empno ;
```

شرط الربط بين الجدولين

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

في المثال السابق تم عرض أرقام الموظفين وأسمائهم من جدول الموظفين(WORKER) كما تم عرض أسماء المدراء من جدول المدراء (MANAGER) عن طريق استخدام الربط الداخلي لنفس الجدول (Self Join) .

الربط بين أكثر من جدولين :

لربط أكثر من جدولين لابد أن تتوفر علاقة ما بينهم جميعاً علماً بأنه لابد أن تكون جمل الشرط المستخدمة في عملية الربط تساوي (عدد الجداول -١) ، أي إذا كان لدينا جدولان فلابد من أن هناك شرط واحد لربطهما ، وإذا كان لدينا ثلاثة جداول فيجب أن يتوفر شرطان لربطهما وهكذا . ولابد من وضع المعامل (AND) بين هذه الشروط .

المثال التالي يوضح كيفية ربط ثلاثة جداول معاً لعرض بيانات من كل منهم .

مثال (٦) :

```
SQL> SELECT e.empno , e.ename , e.sal , d.dname , s.grade
  2  FROM   emp e , dept d , salgrade s
  3  WHERE  e.deptno=d.deptno
  4  AND    e.sal BETWEEN s.losal and s.hisal ;
```

شرط الربط بين جدول الموظفين وجدول الإدارات

شرط الربط بين جدول الموظفين وجدول الفئات

EMPNO	ENAME	SAL	DNAME	GRADE
7369	SMITH	800	RESEARCH	1
7876	ADAMS	1100	RESEARCH	1
7900	JAMES	950	SALES	1
7521	WARD	1250	SALES	2
7654	MARTIN	1250	SALES	2
7934	MILLER	1300	ACCOUNTING	2
7499	ALLEN	1600	SALES	3
7844	TURNER	1500	SALES	3
7566	JONES	2975	RESEARCH	4
7698	BLAKE	2850	SALES	4
7782	CLARK	2450	ACCOUNTING	4
7788	SCOTT	3000	RESEARCH	4
7902	FORD	3000	RESEARCH	4
7839	KING	5000	ACCOUNTING	5

في المثال السابق تم ربط ثلاثة جداول مع بعضها وهي جدول الموظفين (EMP) وجدول الإدارات (DEPT) وجدول الفئات (SALGRADE) وذلك لعرض أرقام الموظفين وأسمائهم ورواتبهم وأسماء الإدارات التابعين لها والفئات التي تنتمي لها رواتبهم .

لاحظ أننا استخدمنا شرطين لربط ثلاثة جداول .

أسئلة الفصل السادس

١ – اكتب جملة استعلام لعرض أسماء وأرقام إدارات وأسم الإدارات للموظفين المسجلين في الإدارة رقم (30) ، بحيث تظهر النتيجة كالتالي :

ENAME	DEPTNO	DNAME
ALLEN	30	SALES
WARD	30	SALES
MARTIN	30	SALES
BLAKE	30	SALES
TURNER	30	SALES
JAMES	30	SALES

٢ – اكتب جملة استعلام لعرض أسماء الوظائف ومكانها للموظفين المسجلين بالإدارة رقم (٣٠) مع منع التكرار في الوظائف . بحيث تظهر النتيجة كالتالي :

JOB	LOC
CLERK	CHICAGO
MANAGER	CHICAGO
SALESMAN	CHICAGO

٣ – اكتب جملة استعلام لعرض أسماء ووظائف وأسماء الإدارات للموظفين المسجلين بالإدارة التي تقع في مدينة (DALLAS) ؟ بحيث تظهر النتيجة كالتالي :

ENAME	JOB	DNAME
SMITH	CLERK	RESEARCH
JONES	MANAGER	RESEARCH
SCOTT	ANALYST	RESEARCH
ADAMS	CLERK	RESEARCH
FORD	ANALYST	RESEARCH

٤ – اكتب جملة استعلام لعرض أسماء ووظائف وأسماء إدارات ورواتب وفئات الرواتب للموظفين المسجلين بالإدارة رقم (10) ؟ بحيث تظهر النتيجة كالتالي :

ENAME	JOB	DNAME	SAL	GRADE
MILLER	CLERK	ACCOUNTING	1300	2
CLARK	MANAGER	ACCOUNTING	2450	4
KING	PRESIDENT	ACCOUNTING	5000	5

٥ – اكتب جملة استعلام لعرض أرقام وأسماء الموظفين وأرقام وأسماء المديرين لهم وذلك للموظفين المسجلين بالإدارة رقم (10) ؟ بحيث تظهر النتيجة كالتالي :

EMPNO	ENAME	MANAGER_NO	MANAGER_NAME
7782	CLARK	7839	KING
7934	MILLER	7782	CLARK