

الفصل الثاني عشر الفهرسة

إذا نظرنا في فهرس كتاب ، نبحث عن موضوع فسيخبرنا الفهرس أن هذا الموضوع ما بين صفحة كذا و صفحة كذا ، عندئذ نذهب إلى الصفحة المعنية مباشرة ، وهكذا تعمل أيضا الفهرسة Indexes مع الجداول في قواعد البيانات، فهي عبارة عن مسار يؤدي مباشرة بسرعة وبسهولة الى موقع البيانات التي نريدها . مع نهاية هذا الفصل سوف تكون تعلمت ما يلي :

- ◆ إنشاء أو تكوين الفهارس Indexes
- ◆ إنشاء أو تكوين مجموعة مركبة من Indexes
- ◆ ما هي وظيفة Indexes ؟
- ◆ أين تستخدم Indexes ؟
- ◆ متى نستخدم Indexes ومتى يستحسن عدم استخدامها ؟

يستخدم الفهرس (Index) طريقة لعرض البيانات تختلف قليلا عن الطريقة التي يتم بها حفظ البيانات على الأقراص الصلبة (وسائط التخزين). بعض أنواع الفهارس Indexes تقوم بإعادة ترتيب البيانات في مواقعها الأساسية داخل الجدول. يتم تكوين الفهرس Index باختيار حقل أو أكثر من جدول. عندما يستخدم Index يتم عرض البيانات بترتيب مختلف. ويمكن التحكم في الترتيب أثناء تكوين Index عندما تستخدم الأمر .create index.

باستخدامك للفهرسة Indexing سوف تلاحظ وجود فارق كبير من حيث حسن الأداء وسرعة الاستجابة و خاصة عندما تستخدم فهرسة Index بغرض دمج بيانات أكثر من جدول.

تستخدم الفهرسة (Indexes) في لغة SQL لثلاث أسباب رئيسية هي:

١. تيسير ترتيب البيانات أو إعادة ترتيبها بناء علي محتويات حقول الفهرسة.
٢. زيادة سرعة تنفيذ Queries إلي السرعة المثالية.
٣. ضبط قواعد البيانات باستخدام المفردات UNIQUE أو Primary Key .

فكرة الفهرسة (Indexes)

عادة يتم التعامل مع البيانات بطريقتين ، الأولى وهي طريقة التتابع و فيها يتم البحث حسب ترتيب البيانات Sequential Access Method و مثال ذلك عندما يكون عندك ملف word مكون من ١٠٠ صفحة و تريد ان تبحث عن كلمة معينة فانك تبدأ في القراءة من أول كلمة ثم التي تليها و هكذا حتى تصل إلى الكلمة المنشودة و إذا كان حظك جيدا "مثلي" فانك ستجد هذه الكلمة في السطور الأخيرة من الصفحة رقم ١٠٠ (إذا وجدتها). اما إذا استخدمت الأمر find الذي يستخدم مع word وتقول له ابحث عن هذه الكلمة ستجد المؤشر في التو واللحظة يشير الى الكلمة ولكن بسرعة الكمبيوتر ولكنه يعمل بالتتابع اي انه يقرأ الملف كلمة كلمة بدلا عنك. ولكن طريقة الوصول المباشر تختلف قليلا. مثلا اذا كنت تعمل في مكتبة مثل دار الكتب وتريد البحث عن كتاب معين

يدخل ضمن موضوعات الهندسة الوراثية ثم أكثر تحديدا الهندسة الوراثية للنباتات. كيف تبحث عن هذا الكتاب وسط عدة ملايين من الكتب.

يوجد في المكتبة ٣٠٠٠ دولار وكل دولار به ٥ أرفف فإذا كان لدى أمين المكتبة كتالوج يقوم أمين المكتبة بتصنيف الكتب فيه ، فانه سيقوم بعملية تصنيف الكتب إلى موضوعات فإذا ذهبت تبحث داخل الكتالوج عن علوم ثم هندسة وراثية ثم نباتات ثم تبحث عن الكتاب الذي تريده، فإذا وجدته ستجد مدون بجانه رقم الدولار ثم رقم الرف. هذا الكتالوج الذي لدي امين المكتب عبارة عن Index يوصلك مباشرة الى موضع الكتاب . وبإضافة Index إلى قواعد البيانات (الجداول) يتمكن SQL من الوصول المباشر للبيانات تسمى هذه الطريقة Direct Access Method .

يستخدم SQL طريقة لتخزين الفهارس Indexes تشبه الكتالوج الذي تحدثنا عنه أو يستخدم ما يعرف بشكل الشجرة Tree-Like Structure وهي بالتالي تستخدم نفس طريقة البحث التي تحدثنا عليها في دار الكتب ولكن بطريقة الكترونية . نظام البحث يشبه الشجرة حيث الفروع الكثيرة لأي شجرة حتى تصل الى ورقه من أوراق الشجرة محل بحثك .

وأود عزيزي القارئ أن تأخذ الأمر ببساطة فان نظام قواعد البيانات يقوم عنك بكل شيء كل ما عليك عمله هو إخباره برغبتك في إعادة ترتيب محتويات الجدول طبقا لبيانات حقل معين . بعبارة أخرى برغبتك في إنشاء Index على حقل من حقول جدول فيقوم بتنفيذ رغبتك

إنشاء فهرس

يتم إنشاء الفهرس (Index) باستخدام أمر CREATE INDEX ويأخذ الشكل العام التالي

```
Create Index index_name on  
table_name(column_name1, column_name2,);
```

حيث يمكن إنشاء Index علي أكثر من حقل Column1 و Column2 وهكذا !

هذه هي صياغة الجملة الشائعة الاستخدام ، ولكن تختلف الإمكانيات والاختيارات فيما بين نظام إدارة قواعد بيانات و آخر ولكن كل هذه النظم تشترك في النقاط الأساسية لتكوين Index فهي على الأقل تتفق في الجملة الرئيسية التالية :

**CREATE INDEX index_name ON table_name
(column_name,)**

سنقوم بإنشاء Index على حقل ما يسمى **account_id** من جدول موجود لدينا يعرف باسم **bills** ، تعال أولاً نرى ما هي محتويات هذا الجدول في الجملة التالية :

select * from bills

Name	Amount	Account_Id
Compu Science	390	10
Egypt Cables	50	11
United Group	634	12
Sara Intl Group	39	10
Basic Solutions	62	10
Computer Technology	37	11
Gama limited	164	15
Byte for IT	14	16
Guide	390	17

لإنشاء فهرس أو Index باسم **ID_INDEX** من جدول **Bills** بناء على محتويات حقل **Account_id** بعبارة أخرى لإعادة ترتيب بيانات جدول **Bills** طبقاً لمحتويات حقل **Account_id** استخدم الجملة الآتية :

create index id_index on bills (account_id)

لاحظ عزيزي القارئ أن الترتيب الذي تم بعد استخدام Index مع **MYSQL** هو ترتيب داخلي لا يظهر مع استخدام جملة **select** السابقة، ولكن مع **SQL Oracle** يظهر الترتيب الحقيقي كما في المثال التالي ولإظهار السجلات طبقاً لترتيب حقل **Account_id**:

SELECT * FROM BILLS;

NAME	AMOUNT	ACCOUNT_ID
Compu Science	345	10
Sara Intl Group	35	10

Basic Solutions	56	10
Egypt Cables	45	11
Computer Technology	34	11
United Group	567	12
Beta for Export	250	13
Gama limited	140	15
Byte for IT	13	16
Guide	345	17

ولكنك إذا قمت بطباعة حقل Account_id الذي عليه Index فسوف تتم الطباعة،
كما يلي :

Select account_id from bills

account_id

10
10
10
11
11
12
13
15
16
17

ولكن ما هو الفرق بين ترتيب السجلات باستخدام أمر INDEX والترتيب باستخدام
الخيار ORDER BY. الفرق بين جدول به حقل Index واستخدام ORDER BY في
جملة select لترتيب الجدول ، هو أن الأول Index قام بترتيب الجدول داخل قاعدة
البيانات بينما جملة select تقوم بترتيب الجدول كل مرة، وهذا يوضح أن عملية الترتيب
الداخلي توفر وقت تنفيذ Select . جرب المثال التالي :

Select * from bills order by account_id

name	amount	account_id
Compu Science	345	10
Sara Intl Group	35	10
Basic Solutions	56	10
Egypt Cables	45	11
Computer Technology	34	11
United Group	567	12
Beta for Export	250	13

Gama limited	140	15
Byte for IT	13	16
Guide	345	17

عندما تقوم بإلغاء جدول فان كل Index سيتم إلغاؤها بالضرورة، فإذا أُلغيت جدول **BILLS** فان Index التي أنشأها في المثال السابق **ID_INDEX** سيتم إلغاؤها أيضا !



ترتيب السجلات تنازليا

الترتيب التلقائي الذي ينشئه أمر **CREATE INDEX** هو الترتيب التصاعدي . وهذا ما لاحظناه في الأمثلة السابقة. فيما يلي سنقوم بإنشاء Index على حقل **AMOUNT** ولكن بحيث يقوم Index بترتيب حقل **AMOUNT** بطريقة تنازلية **Descending**، في هذه الحالة لابد من إضافة الاختيار **DESC** كما في المثال التالي :

```
CREATE INDEX DESC_AMOUNT
ON BILLS(AMOUNT DESC);
Index created.
```

هذه هي المرة الأولى التي نستخدم فيها الترتيب التنازلي **DESCENDING ORDER** وهي تعني أن يقوم **SQL** بترتيب حقل **INDEX** ترتيب تنازلي داخل الجدول **BILLS** في قاعدة البيانات ، سنقوم باختبار الجدول بعد تكوين **Index** :

```
SELECT * FROM BILLS;
```

name	amount	account_id
United Group	567	12
Compu Science	345	10
Guide	345	17
Beta for Export	250	13
Gama limited	140	15
Basic Solutions	56	10
Egypt Cables	45	11
Sara Intl Group	35	10
Computer Technology	34	11
Byte for IT	13	16

كما نلاحظ فان الكميات **AMOUNT** والتي أنشأنا عليها **INDEX** قد تم ترتيبها ترتيب تنازلي باستخدام **DESC** .

هناك عدد من النقاط يجب أخذها في الاعتبار عند استخدام **Index**:

- لن يكون هناك فرق يذكر في مستوى الأداء إذا كانت الجداول صغيرة.
- تقوم **Indexes** بتحسين الأداء بطريقة ملحوظة عندما تستخدم مع حقول بها بيانات متغيرة ومختلفة.
- يمكن اعتبار استخدام **Indexes** مثاليا إذا كانت البيانات المستخرجة كبيرة الحجم.
- استخدام **Indexes** للاستعلام يؤدي إلى نتيجة عكسية خاصة إذا تمت التعديلات في حقول مخصصه للفهرسة **Indexes**.
- تستخدم **Indexes** مساحة إضافية غير تلك التي يستخدمها نظام قواعد البيانات .
- اختر دائما الحقول التي تستخدم كأساس لدمج الجداول **Join**.
- لا تخصص حقول يتم تعديلها بصفة دورية مع **Index**.
- لا يجب أن يحتوي الحقل المستخدم على العديد من القيمة **NULL** .

الفهرسة باستخدام أكثر من حقل.

يمكنك عمل **Index** لأكثر من حقل. هذا النوع من **Index** تسمى **Composite Index** وفيما يلي سوف نقوم بتكوين **Index** لأكثر من حقل .

فيما يلي نقوم بإنشاء **Index** بناء على بيانات حقلين من جدول **bills** وهما حقل **account_id** ثم حقل **amount** . لاحظ أن ترتيب البيانات داخل حقل **account_id** أولا ثم ترتيب البيانات في حقل **amount** تبعاً لرقم الحساب :

```
create Index comp_Index  
on bills (account_id, amount)
```

```
go
```

```
The command(s) completed successfully.
```

```
Select account_id, amount from bills
```

```
Go
```

```
Account_id          Amount
```

```
١٣٧
```

10	39
10	62
10	390
11	37
11	50
12	634
15	164
16	14
17	390

ولكي تحقق اكبر معدل أداء في استخدام Index لأكثر من حقل ، اختر أولاً الحقل الذي تستخدمه أكثر من غيره ثم بعده الحقل الذي يليه في الاستخدام.
في المثال السابق على سبيل المثال رقم الحساب ١٠ تكرر ٣ مرات ثم تلاه رقم الحساب ١١ مرتين ... وهكذا . وهذا أول ترتيب وهو ترتيب أرقام الحسابات. الترتيب الثاني هو المبلغ (amount) فتم ترتيب المبالغ مع رقم الحساب ١٠ كما يلي: ٣٩ ثم ٦٢ ثم ٣٩٠.

عرفنا أن composite Index عبارة عن استخدام أكثر من حقل عند تكوين Index. و إذا كان مستوى الأداء و سرعته عامل هام لديك فكر في استخدام Index لحقل واحد single Index.



استبعاد التكرارات من الفهرسة

عادة يستخدم Index مع الحقول الغير متكررة UNIQUE وذلك لمنع ظهور قيم متكررة لنفس البيانات. فعلى سبيل المثال دعنا نضع القاعدة التالية كل فاتورة يتم دفعها بواسطة شركة ما يجب أن تأتي من حساب بنك مختلف، هنا يجب إنشاء Index غير متكررة UNIQUE على كل من حقل name وحقل Account_Id استخدم الخيار UNIQUE لهذا الغرض.

إذا قمت بتحديد حقل على انه Primary Key يقوم نظام قواعد البيانات تلقائياً بإنشاء Index لان كل قيم حقل Primary Key غير متكررة UNIQUE.



حذف الفهرس

يستخدم لحذف الفهرس أمر **DROP INDEX** بالشكل الآتي :

DROP INDEX index_name ;

