قواعد البيانات

SQL

SQL SELECT

- إن الاستخدام الرئيسي للغة SQL هو لاختيار البيانات الموجودة في الجداول . سنستخدم كلمتين أساسيتين لاستخراج البيانات وهى **SELECT**و **FROM** .

- SELECT "column_name"
- FROM "table_name"

اختار أسماء كل المتاجر في هذا الجدول

Store_Information

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999
Los Angeles	$300	Jan-08-1999
Boston	$700	Jan-08-1999

```
SELECT    store_name
FROM      Store_Information
```

وتكون نتيجة الاستعلام:

store_name
Los Angeles
San Diego
Los Angeles
Boston

DISTINCT

- SELECT
  - تسمح لنا باستخراج كل المعلومات من عمود أو عدة أعمدة في جدول ما.
  - هذا بالطبع يعنى أمكانية تكرار بعض البيانات ماذا يحدث إذا كنا نرغب فقط في اختيار العناصر الغير متكررة؟

- SELECT DISTINCT "column_name"
- FROM "table_name"

DISTINCT

- **SELECT DISTINCT store_name**
- **FROM Store_Information**

وتكون نتيجة الاستعلام:

store_name
Los Angeles
San Diego
Boston

WHERE

والآن فقد نرغب في اختيار البيانات من جدول ما حسب شروط معينة.
استرجاع أسماء المحلات التي قامت بمبيعات أكثر من ألف دولار:

```
SELECT      "column_name"
FROM        "table_name"
WHERE             "condition"
```

وتكون النتيجة كما يلى :-

```
SELECT      store_name
FROM        Store_Information
WHERE       Sales > 1000
```

store_name
Los Angeles

SQL OR AND


```
SELECT
"column_name"
FROM          "table_name"
WHERE         "simple condition"
{[AND|OR] "simple condition"}
```

- هذا الشرط من الممكن أن يكون بسيطا مثل المستخدم في المثال السابق ومن الممكن أيضا أن يكون شرطا مركبا. الشروط المركبة مكونة من مجموعة من الشروط البسيطة المرتبطة بكلمتى OR أو AND ليس هناك حد أقصى لعدد الشروط البسيطة التي من الممكن تواجدها في عبارة واحدة لـ SQL .

SQL OR AND

على سبيل المثال نستطيع اختيار كل المتاجر التي حققت مبيعات أكثر من ألف دولار وكل المحلات التي حققت أقل من 500 دولار وأكثرا من 275 دولار من الجدول السابق. شكل عبارة الـ SQL يكون كما يلى :-

```
SELECT      store_name
FROM        Store_Information
WHERE             Sales > 1000
OR          (Sales < 500 AND Sales > 275)
```

وتكون نتيجة الاستعلام

store_name
Los Angeles
San Francisco

SQL IN

تستخدم عند معرفة القيم التي تبحث فيها عن عمود معين بدقة.

أن عدد القيم المسموح بوضعها في الاقواس من الممكن أن تكون قيمة أو أكثر مع فصل كل القيم باستخدام الفاصلة.
نستطيع استخدام قيم عددية أو حروف.

```
SELECT      "column_name"
FROM        "table_name"
WHERE       "column_name"       IN
('value1', 'value2', ...)
```

SQL IN

على سبيل المثال إذا أردنا اختيار كل السجلات المقابلة لمدينتى لوس أنجلوس وسان دييجو في الجدول التالي.

```
SELECT        *
FROM   Store_Information

WHERE  store_name IN ('Los Angeles','San Diego')
```

ونتيجة الاستعلام كما يلى :-

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
San Diego	$250	Jan-07-1999

SQL BETWEEN

تسمح باختيار قيمة في مدي معين

```
SELECT          "column_name"
FROM            "table_name"
WHERE           "column_name"
BETWEEN 'value1' AND 'value2'
```

وبذلك سوف نختار كل الاعمدة التي تقع قيمة العمود فيها بين الرقمين VALUE 1 و VALUE 2.

SQL BETWEEN

على سبيل المثال فإذا رغبنا في اختيار كل بيانات المبيعات من 6 يناير 1999 الى 10 يناير 1999 من جدول بيانات المحلات.

```
SELECT              *
FROM          Store_Information
WHERE         Date BETWEEN 'Jan-06-1999'
AND          'Jan-10-1999'
```

store_name	Sales	Date
San Diego	$250	Jan-07-1999
San Francisco	$300	Jan-08-1999
Boston	$700	Jan-08-1999

SQL LIKE


```
SELECT      "column_name"
FROM        "table_name"
WHERE       "column_name" LIKE {PATTERN}
```

تسمح بصورة أساسية بأن تقوم بعمل بحث مبنى على شكل معين بدلا من أن نحدد بدقة الكلمات التي نرغب في البحث ضمنها.

SQL LIKE


```
SELECT              *
FROM          Store_Information
WHERE         store_name   LIKE '%AN%'
```

وتكون نتيجة الاستعلام

store_name	Sales	Date
LOS ANGELES	$1500	Jan-05-1999
SAN FRANCISCO	$300	Jan-08-1999
SAN DIEGO	$250	Jan-07-1999

SQL ORDER BY


```
SELECT        "column_name"
FROM          "table_name"
[WHERE        "condition"]
ORDER BY      "column_name" [ASC, DESC]
```

- حتى الآن فقد اختبرنا كيف يمكن أن نختار بيانات من جدول باستخدام SELECT و WHERE .
- غالبا ما نحتاج إلى ترتيب النتائج بصورة معينة. من الممكن ترتيب البيانات ترتيبا تصاعديا (ascending) أو تنازليا (descending) بناء على قيمة عمود مخزن به أرقام أو حروف. في هذه الحالة نستخدم كلمة ORDER BY لتحقيق هذا الهدف .

SQL ORDER BY

- على سبيل المثال إذا أردنا الحصول على قائمة بمحتوى جدول بيانات المحلات مرتبة تنازليا طبقا لقيمة المبيعات (SALES) فأن عبارة SQL التي تحقق ذلك هى كما يلى:

```
SELECT      *
FROM        Store_Information
ORDER BY    Sales DESC
```

store_name	Sales	Date
Los Angeles	$1500	Jan-05-1999
Boston	$700	Jan-08-1999
San Francisco	$300	Jan-08-1999
San Diego	$250	Jan-07-1999

SQL GROUP BY

إن كلمة GROUP BY تستخدم عندما نقوم باختيار أعمدة متعددة من جدول أو أكثر من جدول ويكون هناك على الأقل عملية رياضية موجودة في عبارة SELECT. عندما يحدث ذلك نحن نكون بحاجة إلى أن نقوم بتجميع (GROUP BY) البيانات بناء على الأعمدة التي قمنا باختيارها وهى كل الأعمدة المذكورة فيما عدا الأعمدة المستخدمة في العمليات الحسابية.

```
SELECT "column_name1", SUM("column_name2")
FROM          "table_name"
GROUP BY      "column_name1"
```

SQL GROUP BY


```
SELECT      store_name, SUM(Sales)
FROM        Store_Information
GROUP BY    store_name
```

store_name	SUM(Sales)
Los Angeles	$1800
San Diego	$250
Boston	$700

SQL HAVING


```
SELECT      "column_name1", SUM("column_name2")
FROM        "table_name"
GROUP BY    "column_name1"
HAVING      (arithematic function condition)
```

هناك شئ آخر قد نحتاج اليه عند أجراء العمليات الحسابية وهو أن يقتصر الناتج على المجموعات التي تحقق شرطا معينا مثلا مجموع معين (أو أى عملية حسابية أخرى).

على سبيل المثال من الممكن أن نرغب في عرض نتائج المحلات التي تتعدي مبيعاتها 1500 دولار فقط.

SQL HAVING


```
SELECT      store_name, SUM(sales)
FROM        Store_Information
GROUP BY    store_name
HAVING      SUM(sales) > 1500
```

store_name	SUM(Sales)
Los Angeles	$1800

SQL ALIAS

الأسماء البديلة (ALIASES).
هناك نوعان من الكلمات البديلة يستخدمان مع لغة SQL وهما الاسم البديل للعمود و الاسم البديل للجدول.
استخدام أسم بديل للعمود يساعد كثيرا في جعل نتيجة الاستعلام أكثر قبولا.
النوع الثاني مناسبا جدا للحصول على معلومات من جدولين منفصلين (تسمى تلك بعملية ربط الجداول).

```
SELECT   "table_alias"."column_name1"
"column_alias"
FROM    "table_name" "table_alias"
```

SQL ALIAS


```
SELECT      A1.store_name Store, SUM(A1.Sales)  "Total Sales"
FROM        Store_Information   A1
GROUP BY    A1.store_name
```

Store	Total Sales
Los Angeles	$1800
San Diego	$250
Boston	$700

Summary


```
SELECT [ALL | DISTINCT]
column1[,column2]

FROM   table1[,table2]

[WHERE      "conditions"]

[GROUP BY   "column-list"]

[HAVING     "conditions"]

[ORDER BY   "column-list" [ASC|DESC]]
```

علامات المقارنة

=	Equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
<> or !=	Not equal to
LIKE	String comparison test

الدوال التجميعية

MIN	تحسب القيمة الصغرى للأرقام في عمود ما.
MAX	تحسب القيمة العظمى للأرقام في عمود ما.
SUM	تحسب مجموع قيم الأرقام في عمود ما.
AVG	تحسب متوسط قيم الأرقام في عمود ما.
COUNT	تحسب عدد الأرقام في عمود ما.
COUNT(*)	تحسب عدد الصفوف في جدول ما.

الدوال التجميعية


```
SELECT      "function type"("column_name")
FROM        "table_name"
```

علي سبيل المثال إذا رغبنا في الحصول علي مجموع كل مبيعات الشركة

```
SELECT      SUM(Sales)
FROM        Store_Information
```

SUM(Sales)
$2750

الدوال التجميعية

الاستعلام التالي يحسب متوسط مرتبات الموظفين.

```
SELECT    AVG(salary)
FROM      employee
```

والاستعلام التالي يحسب مرتبات الموظفين والذين يعملون كمبرمجين.

```
SELECT    AVG(salary)
FROM      employee
WHERE     title = 'Programmer'
```

SQL COUNT

هى دالة رياضية أخرى وهي تسمح لنا بالحصول علي عدد الصفوف في جدول معين

```
SELECT      COUNT("column_name")
FROM        "table_name"
```

SQL COUNT

الحصول علي عدد المحلات في الجدول

```
SELECT      COUNT(store_name)
FROM        Store_Information
```

وتكون نتيجة الاستعلام

Count(store_name)
4

SQL COUNT

كلمة COUNT وكلمة DISTINCT من الممكن استخدامهما معا في عبارة واحدة للحصول علي عدد الحقول الغير متكررة في اى جدول.

علي سبيل المثال إذا أردنا إيجاد عدد المحلات بدون تكرار أي منهم فسنكتب العبارة التالية:

```
SELECT      COUNT(DISTINCT store_name)
FROM        Store_Information
```

وتكون نتيجة الاستعلام

Count(DISTINCT store_name)
3

IN & BETWEEN


```
SELECT    col1, SUM(col2)
FROM      list-of-tables
WHERE     col3 IN  (list-of-values)
```

إذا كان معامل IN يستخدم لاختيار الصفوف التي تحقق شرط التواجد في فئة معينة من الارقام.
يمكننا أيضا استخدام المعامل (NOT IN) لاختيار الصفوف التي لا تكون القيم بها داخل هذه القائمة وبالتالي فاننا نقوم باستبعاد القيم داخل القائمة.

IN & BETWEEN

BETWEEN

يستخدم لاختبار إذا كانت القيمة تقع بين قيمتين

هذه العبارة تقوم باختيار بيانات الموظف الذى يقع عمره بين 30 و 40
مع ملاحظة أن الرقمين 30 و 40 من ضمن الارقام المطلوبة.

```
SELECT    eid, age, last_name, salary
FROM      employee_info
 WHERE    age   BETWEEN 30 AND 40
```