الأكاديمية العربية الدولية
Arab International Academy

إسم المادة: لغة الاستعلامات الهيكلية

إسم الدكتور: حسام يونسو

الأكاديمية العربية الدولية – منصة أعد

محاور المادة

- المقدمة
- تعريف لغة الإستعلامات الهيكيلية (SQL)
- أهمية SQLفي إدارة قواعد البيانات
- الاستعلامات (Queries)
- العمليات (Transactions)
- التحكم في البيانات (Data Control)
- الأمر SELECT
- الأمر INSERT
- الأمر UPDATE
- الأمر DELETE
- جملة WHEREفي SQL
- عوامل المقارنة في WHERE
- العوامل المنطقية في WHERE
- إنشاء الجداول CREATE TABLE
- القيود (Constraints)
- نصائح عند إنشاء الجداول
- تعديل الجداول ALTER TABLE
- استخدامات أخرى لجملة ALTER TABLE
- حذف الجداول DROP TABLE
- أفضل الممارسات لتحسين كفاءة الاستعلامات
- استخدامات SQL في البرمجة وتطوير التطبيقات
- التفاعل مع البيانات عبر الويب
- نظم إدارة قواعد البيانات المشهورة
- أهمية التعلم والتطبيق المستمر

المقدمة

في عالم اليوم المليء بالبيانات، أصبحت **لغة الاستعلامات الهيكلية (SQL)** الأداة الأساسية لإدارة البيانات داخل قواعد البيانات العلائقية .سواء كنت تقوم بتخزين بيانات العملاء، أو تحليل النتائج المالية، أو تشغيل تطبيقات الويب، فإن SQL توفر الوسائل اللازمة لإدارة البيانات بشكل فعال وآمن.

خلال هذا العرض، سنتعرف على المبادئ الأساسية لـSQL، بما في ذلك كيفية إنشاء الجداول، تعديلها، وحذفها، وكذلك أفضل الممارسات لتحسين أداء الاستعلامات وضمان سلامة البيانات .إن إتقان هذه اللغة يفتح الباب أمام بناء تطبيقات قوية وموثوقة، مع الاستفادة القصوى من البيانات المتاحة.

تعريف لغة الإستعلامات الهيكلية (SQL)

ما هي SQL؟

SQL (Structured Query Language) هي لغة البرمجة المستخدمة لإدارة واستعلام البيانات في نظم إدارة قواعد البيانات العلائقية.

تم تطوير SQL في السبعينيات، وهي تتبع المعايير العالمية (ANSI).

تتيح SQL للمستخدمين إجراء عمليات على البيانات مثل الاستعلام، والإضافة، والتحديث، والحذف.

أهمية SQL في إدارة قواعد البيانات

التفاعل مع البيانات: توفر SQL وسيلة فعالة لاسترجاع البيانات وإجراء عمليات عليها من قواعد البيانات.

إدارة البيانات الكبيرة: تدعم SQL التعامل مع كميات كبيرة من البيانات، مما يجعلها مناسبة للتطبيقات التجارية والمشاريع الكبرى.

الأمان: توفر SQL آليات للتحكم في الوصول إلى البيانات، مما يساعد في حماية المعلومات الحساسة.

التوحيد: تستخدم SQL كمعيار صناعي، مما يعني أن المهارات المكتسبة في SQL يمكن تطبيقها على مجموعة متنوعة من نظم إدارة قواعد البيانات.

التقارير والتحليل: يمكن استخدام SQL لإنشاء تقارير تحليلية، مما يساعد الشركات في اتخاذ قرارات مستنيرة.

الدعم المجتمعي: تمتلك SQL مجتمعًا واسعًا من المطورين والمستخدمين، مما يسهل العثور على الموارد والدروس والدعم.

الاستعلامات (Queries)

الاستعلامات (Queries):

التعريف:الاستعلامات هي أوامر تستخدم لاسترجاع البيانات من قاعدة البيانات.

الأنواع:

- **استعلامات الاختيار (SELECT)**: تُستخدم لاسترجاع البيانات.
  - مثال:

```
SELECT * FROM employees WHERE position = 'Developer';
```

استعلامات التجميع (Aggregate Queries): تُستخدم لتجميع البيانات وإجراء عمليات حسابية عليها.

  - مثال:

```
SELECT COUNT(*) FROM employees WHERE department = 'Sales';
```

العمليات (Transactions)

العمليات هي مجموعة من الأوامر التي تُنفذ كوحدة واحدة .إذا فشلت أي عملية في التنفيذ، يتم التراجع عن جميع العمليات.

الخصائص :تضمن العمليات سلامة البيانات من خلال خصائص :ACID

Atomicity (الذرية :(يجب أن تُنفذ جميع العمليات أو لا تُنفذ على الإطلاق.

Consistency (التناسق :(يجب أن تظل قاعدة البيانات في حالة متناسقة بعد تنفيذ العمليات.

Isolation (العزل :(يجب أن تكون العمليات معزولة عن بعضها البعض.

Durability (الاستدامة :(يجب أن تبقى التغييرات دائمة بعد تأكيد العملية.

أمثلة:
تنفيذ عملية نقل الأموال بين حسابين

```
BEGIN TRANSACTION;
UPDATE accounts SET balance = balance - 100 WHERE account_id = 1;
UPDATE accounts SET balance = balance + 100 WHERE account_id = 2;
COMMIT;
```

التحكم في البيانات (Data Control)

التحكم في البيانات يتعلق بتأمين البيانات والتحكم في من يمكنه الوصول إليها.

أهمية التحكم في البيانات:

- يحمي المعلومات الحساسة.
- يضمن أن المستخدمين يمكنهم فقط الوصول إلى البيانات التي يحتاجون إليها لأداء مهامهم.

الأوامر الرئيسية:

GRANT: تُستخدم لمنح حقوق الوصول للمستخدمين.

- مثال:

```
GRANT SELECT ON employees TO user1;
```

REVOKE: تُستخدم لسحب حقوق الوصول من المستخدمين.

مثال:

```
REVOKE SELECT ON employees FROM user1;
```

الأمر SELECT

وظيفتها :استرجاع البيانات من قاعدة البيانات.

الاستخدام :تقوم بتحديد الأعمدة التي ترغب في استرجاعها من جدول معين بناءً على شروط محددة أو بدون شروط.

مثال:

```
SELECT name, date_of_birth FROM students;
```

في هذا المثال، يقوم الاستعلام باسترجاع **الأسماء** و **تواريخ الميلاد** من جدول الطلاب .سيقوم بإعادة جميع الطلاب في الجدول مع عرض الأسماء وتواريخ الميلاد فقط.

الأمر INSERT

وظيفتها :إضافة سجلات جديدة إلى الجداول.

الاستخدام :يُستخدم لإدخال بيانات جديدة إلى جدول معين من خلال تحديد الأعمدة والقيم المرتبطة بها.

مثال:

```
INSERT INTO students (student_id, name, date_of_birth, email) VALUES (1, 'Ahmed Ali', '2000-01-01', 'ahmed@example.com');
```

هنا، نقوم بإضافة طالب جديد إلى جدول الطلاب مع **معرّف** الطالب (1)، **الاسم** ("Ahmed Ali")، **تاريخ الميلاد** -'2000-01-01') و**البريد الإلكتروني** .('ahmed@example.com')

الأمر UPDATE

وظيفتها :تحديث البيانات الموجودة في الجداول.

الاستخدام :تُستخدم لتحديث سجل موجود بالفعل في الجدول .عادةً ما يكون الأمر مصحوبًا بشروط لتحديد السجل الذي سيتم تحديثه.

مثال:

```
UPDATE students SET email = 'ahmed_new@example.com' WHERE student_id = 1;
```

في هذا المثال، يتم تحديث **البريد الإلكتروني** للطالب الذي لديه **student_id** يساوي 1. سيتم استبدال البريد الإلكتروني القديم بالبريد الجديد ('ahmed_new@example.com').

الأمر DELETE

وظيفتها :حذف سجلات من الجداول.

الاستخدام :تُستخدم لحذف سجلات معينة من الجدول بناءً على شروط محددة.

مثال:

```
DELETE FROM students WHERE student_id = 1;
```

هنا، سيتم حذف الطالب الذي يحمل **student_id** يساوي 1 من جدول الطلاب .يجب توخي الحذر عند استخدام هذا الأمر لضمان حذف السجلات الصحيحة فقط.

جملة WHERE في SQL

WHEREهي جملة تستخدم لتحديد شروط معينة يجب أن تتوافق معها البيانات المُسترجعة أو المُعالجة في استعلامات SQL.

تُستخدم **WHERE**بشكل شائع مع أوامر **SELECT**، **UPDATE**، و **DELETE**لتصفية النتائج.

استخدام WHEREمع SELECT

```
SELECT * FROM employees WHERE position = 'Developer';
```

عوامل المقارنة في WHERE

يمكن استخدام عدة عوامل مقارنة في جملة WHERE، بما في ذلك:

) =يساوي(

<>أو) =!لا يساوي(

) <أقل من(

) >أكبر من(

) =<أقل من أو يساوي(

) =>أكبر من أو يساوي(

العوامل المنطقية في WHERE

يمكن أيضًا دمج الشروط باستخدام عوامل منطقية:

AND: يتطلب أن تكون جميع الشروط صحيحة.

OR: يتطلب أن يكون أحد الشروط صحيحًا.

NOT: يستخدم لنفي شرط معين.

مثال:

```sql
SELECT * FROM employees WHERE position = 'Developer' AND salary > 50000;
```

في هذا المثال، يتم استرجاع جميع الموظفين الذين يحملون وظيفة "Developer" ولديهم راتب أكبر من 50000.

إنشاء الجداول CREATE TABLE

تعريف CREATE TABLE: تُستخدم جملة CREATE TABLE لإنشاء جدول جديد في قاعدة البيانات.

يتضمن الجدول مجموعة من الأعمدة (Fields) وكل عمود له نوع بيانات محدد.

الصيغة العامة:

```
CREATE TABLE table_name (
    column1 datatype [constraints],
    column2 datatype [constraints],
    ...
);
```

مثال عملي

إنشاء جدول الموظفين (employees):

```
CREATE TABLE employees (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(50) NOT NULL,
    position VARCHAR(50),
    hire_date DATE,
    salary DECIMAL(10, 2)
);
```

القيود (Constraints)

القيود هي قواعد تُطبق على الأعمدة في الجداول لضمان صحة البيانات وسلامتها.

تساعد في التحكم في نوع البيانات التي يمكن إدخالها في قاعدة البيانات.

أنواع القيود الأساسية:

NOT NULL: يضمن عدم وجود قيم فارغة في العمود.

UNIQUE: يضمن أن جميع القيم في العمود فريدة ولا تتكرر.

PRIMARY KEY: يحدد عمودًا أو مجموعة من الأعمدة كمفتاح أساسي للجدول.

FOREIGN KEY: يُستخدم لإنشاء علاقة بين جدولين، حيث يشير إلى عمود في جدول آخر.

نصائح عند إنشاء الجداول

قم بتخطيط الهيكل :من المهم أن تقوم بتخطيط هيكل الجدول قبل إنشائه, يساعد التخطيط الجيد في تقليل الحاجة إلى إجراء تعديلات لاحقًا، مما يوفر الوقت والجهد.

استخدم أسماء واضحة :اختر أسماء أعمدة وجداول تعبر بوضوح عن محتواها.

سهولة الفهم :تجعل الأسماء الواضحة من السهل على المطورين والمستخدمين فهم الهيكل وبيانات الجدول دون الحاجة إلى تفسير إضافي .

مثال :استخدم `employee_name` بدلاً من `name` فقط لزيادة الوضوح.

اختر نوع البيانات المناسب :حدد نوع البيانات لكل عمود وفقًا لنوع المعلومات التي ستخزنها, يساعد اختيار النوع المناسب في تجنب الأخطاء في البيانات وضمان سلامتها.

مثال :استخدم `VARCHAR` للنصوص، `INT` للأعداد، و `DATE` للتواريخ.

تعديل الجداول ALTER TABLE

تُستخدم جملة **ALTER TABLE** لتعديل هيكل جدول موجود في قاعدة البيانات.

يمكنك إضافة أعمدة جديدة، تعديل أعمدة موجودة، أو حذف أعمدة.

مثال عملي :إضافة عمود جديد

في هذا المثال، نضيف عمودًا جديدًا يسمى `salary` من نوع عدد عشري (DECIMAL) بحد أقصى 10 أرقام و 2 أرقام عشرية إلى جدول `employees`.

```
ALTER TABLE employees ADD salary DECIMAL(10, 2);
```

استخدامات أخرى لجملة ALTER TABLE

تعديل نوع البيانات لعمود: هنا نقوم بتغيير نوع البيانات للعمود salary إلى عدد صحيح (INT).

```
ALTER TABLE employees MODIFY salary INT;
```

حذف عمود: هذا الأمر يحذف العمود salary من جدول employees.

```
ALTER TABLE employees DROP COLUMN salary;
```

حذف الجداول DROP TABLE

تُستخدم جملة **DROP TABLE** لحذف جدول بالكامل من قاعدة البيانات ,عند استخدام هذه الجملة، يتم حذف الجدول وجميع البيانات المخزنة فيه بشكل دائم.

مثال عملي :حذف جدول ,في هذا المثال، نقوم بحذف جدول employees وكل البيانات المرتبطة به من قاعدة البيانات.

```
DROP TABLE employees;
```

تحذير :تأكد من أنك لا تحتاج إلى الجدول أو البيانات قبل حذفه، لأن العملية لا يمكن التراجع عنها.

التأثيرات على المفاتيح الأجنبية :إذا كان الجدول مرتبطًا بجداول أخرى من خلال مفاتيح أجنبية، قد يتسبب حذف الجدول في حدوث أخطاء إذا لم يتم التعامل معها بشكل صحيح.

أفضل الممارسات لتحسين كفاءة الاستعلامات

تجنب الاستعلامات الثقيلة

- **تحديد نطاق البيانات** :استخدم جملة **WHERE** لتقليل عدد الصفوف المسترجعة إلى الحد الأدنى .كلما كانت البيانات المسترجعة أقل، كانت العملية أسرع.

تجنب استخدام :* SELECT بدلاً من استرجاع جميع الأعمدة، حدد الأعمدة التي تحتاج إليها فقط.

مثال:

```
SELECT name, position FROM employees WHERE salary > 50000;
```

استخدامات SQL في البرمجة وتطوير التطبيقات

إدارة قواعد البيانات SQL :هي اللغة الأساسية لإدارة قواعد البيانات العلائقية .(RDBMS)

تُستخدم في **إضافة، تعديل، استرجاع**، و **حذف** البيانات في الجداول.

تُعد SQLجزءًا أساسيًا من أي تطبيق يعتمد على قاعدة بيانات مثل التطبيقات المالية، أنظمة إدارة المحتوى (CMS)، والتطبيقات الطبية.

التفاعل مع البيانات عبر الويب

تُستخدم SQL في التطبيقات الويب الديناميكية، حيث يتم **التفاعل مع البيانات** من خلال استعلامات SQL للتعامل مع قواعد البيانات.

أمثلة على استخدامات SQL:

تخزين بيانات المستخدمين.

استرجاع النتائج بناءً على طلبات بحث المستخدمين.

تعديل بيانات الحسابات أو السلع.

نظم إدارة قواعد البيانات المشهورة

MySQL: هو أحد أشهر أنظمة إدارة قواعد البيانات العلائقية مفتوحة المصدر.

يُستخدم على نطاق واسع في تطبيقات الويب، خاصةً مع **PHP** و**Node.js**، مثل أنظمة إدارة المحتوى ومنصات التجارة الإلكترونية.

PostgreSQL: هو نظام إدارة قواعد بيانات علائقية مفتوح المصدر ومتقدم يدعم العديد من المميزات، مثل **دعم المعاملات المتقدمة**، و **الامتدادات**.

يُستخدم في تطبيقات تحتاج إلى موثوقية واستقرارية عالية، مثل التحليلات المالية.

SQL Server: هو نظام إدارة قواعد بيانات علائقية تم تطويره من قبل **Microsoft**.

يُستخدم بشكل رئيسي في المؤسسات التي تعتمد على **تقنيات Microsoft**، مثل التطبيقات التجارية والمالية الكبيرة.

أهمية التعلم والتطبيق المستمر

SQLهي لغة قابلة للتطور، وكلما تعمقت في تعلمها وتطبيقها في مشاريعك البرمجية، زادت قدرتك على تحسين الأداء وإدارة البيانات بكفاءة أكبر.

يُوصى بممارسة **الاستعلامات المعقدة** وتحليل البيانات باستمرار للحفاظ على مهاراتك وتعزيز معرفتك.

تبنّي **أفضل الممارسات** في تصميم قواعد البيانات يُحسن أداء التطبيقات ويوفر الكثير من الجهد في المستقبل.

شكراً لكم