وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويـت

تقنية المعلومات 11

الصـف الحادي عشر

الفصل الدراسي الثاني

المرحلة الثانوية

وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويـت

تقنية المعلومات 11

إشراف

أ. منى سالم عوض سالم (رئيس اللجنة)

أ. رأفـــت صابـــر عبـــداللاه أحمـــد

أ. حســـام الديـــن علـــي عبدالقـــادر

أ. أحـــــمد محـــمد عبد الله عيـسى

د.أشـــرف رضـــوان رضـــوان سليـمان

د.يوســف منصـور يوســف الخليفـي

أ. منار مصطفـــى عبدالحميد جمال

أ. منى مـــرزوق مخـلد العـــــازمي

أ. إبراهيم عـبدالله إبراهيم الميـاس

تصميم

أ. ســاره ياســين عبــد الله الأميــر

إخراج

د. أشــرف رضــوان رضــوان سليمان

الطبعة الأولى
1447 هـ
2026/2025 م

الطبعة الأولى 2026/2025م

المراجعة العلمية

أ. بـدور عـــباس حســين بوعـــباس

أ. فـــاطمة نجـــم جـــاسم الهـــولي

أ. غيـــداء حســـن عبدالله السـبتي

أ. محمد عـبد الرزاق ملا محمد علي

الفريق المُساند

أ. سنية محمد علي المؤمن - تصميم

أودع بمكتبة الوزارة تحت رقم (438) بتاريخ 2025/12/29م


بسم الله الرحمن الرحيم

حضرة صاحب السمو الشيخ مشعل الأحمد الجابر الصباح

أمير دولة الكويت

H.H. Sheikh Meshal AL-Ahmad Al-Jaber Al-Sabah
Amir Of The State Of Kuwait

سمو الشيخ صباح خالد الحمد الصباح
ولي عهد دولة الكويت

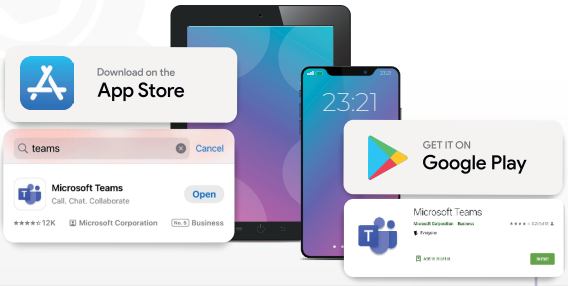
H. H. Sheikh Sabah Khaled Al-Hamad Al-Sabah
Crown Prince Of The State Of Kuwait

الفصل الرقـــمي Digital Classroom

أولًا :تحميل وتثبيت تطبيق Microsoft Teams

الدخول على متجر تطبيقات الأجهزة الرقمية.

من جهاز الحاسوب



من الأجهزة الذكية

ثانياً: تفعيل Microsoft Teams على الجهاز الرقمي

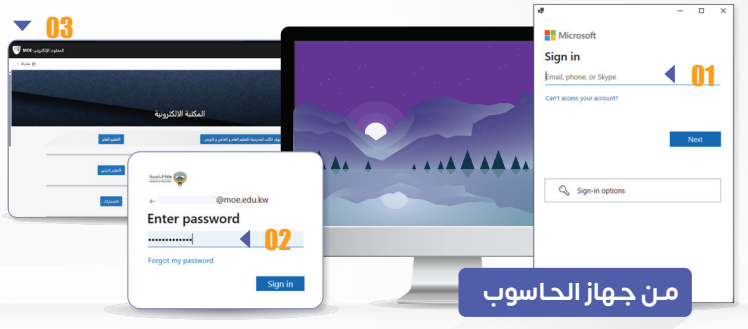
لتشغيل التطبيق اتبع الخطوات التالية:

احرص على تحديث تطبيق Microsoft Teams بشكل دوري.

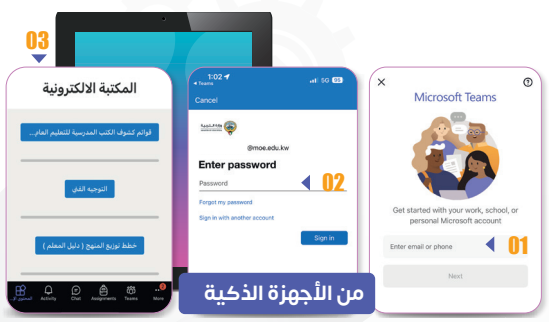
01 أدخل اسم المستخدم: البريد الإلكتروني الرسمي الخاص بحساب Teams.

02 أدخل كلمة المرور الخاصة بحساب Teams.

03 تنقل بين واجهات التطبيق مثل المحتوى الإلكتروني، وفرق المواد الدراسية.



من جهاز الحاسوب



من الأجهزة الذكية

احتفظ بكلمة المرور في مكان آمن لتسهيل تسجيل الدخول لاحقًا.

لا تشارك بياناتك مع أي شخص غير موثوق به.

مـن هـو حـمد ؟...

هـو مُساعد تعليمـي رقمـي ذكـي يعتمـد علـى تقنيـات الـذكاء الاصطناعـي، مُصمـم لتقديـم الدعـم التعليمـي التفاعلـي لكافـة المناهـج الدراسـية.

حمد دائمًا معك... لتتعلّم بثقة وتنجح بامتياز.

مع حـمد Chat

مـا هـي خـدمـة حـمـد شـات؟

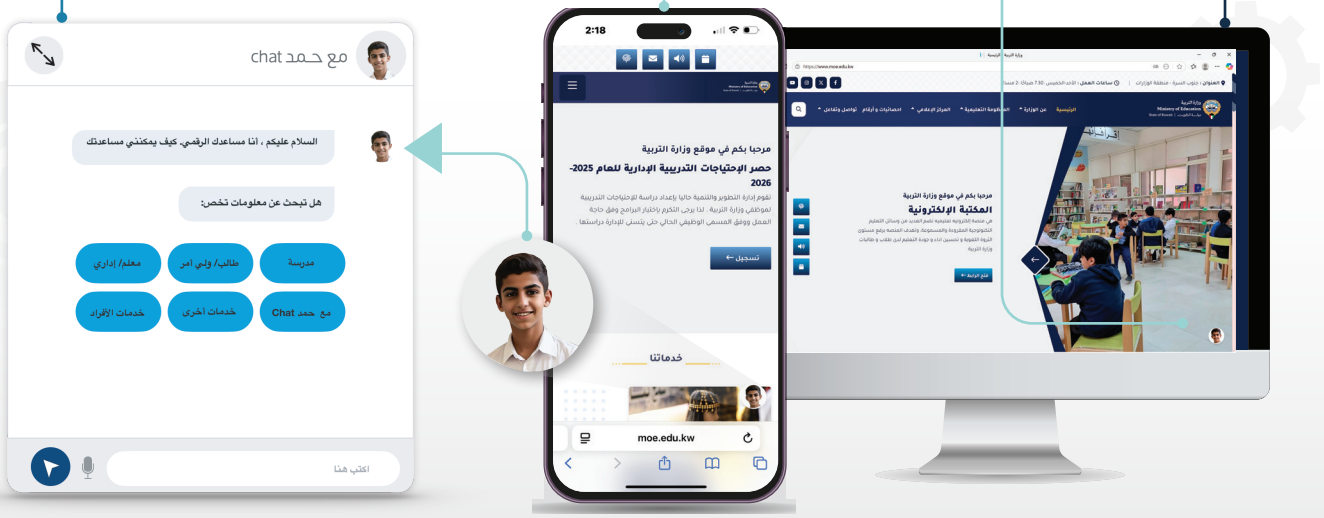
هـي خدمـة المحادثـة الذكيـة المقدمـة مـن وزارة التربيـة فـي دولـة الكويـت، تعتمـد علـى الـذكاء الاصطـناعي، تتمثـل وظيفتـه الأساسـية فـي تسـهيل عمليـة الفهـم والاسـتيعاب مـن خلال تقديـم الشـروحات المبسـطة، والإجابـة علـى الاستفسـارات، وتوجيـه المتعلميـن خلال مسـيرتهم التعليميـة.

أيـــن أجـــده؟

01 - زيارة الموقع الرسمي www.moe.edu.Kw - أو تطبيق وزارة التربية

02 الضغط علـى صـورة حمد شات

03 اكتب استفساراتك، وستتلقى الإجابات بشكل فوري.



المحـــتوى

المقدمة

في ظلّ النمو المتسارع في حجم البيانات وتنوّع مصادرها، أصبحت قواعد البيانات من الركائز الأساسية للأنظمة الرقمية الحديثة، لما توفره من تنظيم فعّال للمعلومات، وسهولة استرجاعها، وضمان سلامتها وترابطها المنطقي، ويُسهم هذا التنظيم في رفع جودة البيانات ودقتها، ودعم اتخاذ القرارات.

ويتناول هذا الكتاب مفهوم العلاقات في قواعد البيانات العلائقية Relational Database، والذي يُسهم في تنظيم البيانات وتقليل التكرار، وضمان سلامتها واتساقها. بما يسمح بتمثيل الارتباطات بين الكيانات المختلفة.

ويركّز الكتاب على إنشاء قواعد البيانات والجداول المترابطة، وتنفيذ عمليات الإدخال والاستعلام والتحديث والحذف CRUD، مع الالتزام بقواعد سلامة البيانات.

كما يسلّط الكتاب الضوء على الذكاء الاصطناعي بوصفه الأساس في تطوير الحلول الرقمية الحديثة، مع إبراز الدور الجوهري لقواعد البيانات العلائقية في دعمه، إذ تعتمد أنظمة الذكاء الاصطناعي على بيانات منظّمة ودقيقة في عمليات التخزين، والتدريب، والتحليل، واتخاذ القرار.

وقد أُعدّ هذا الكتاب ليكون دليلاً تعليميًا مبسّطًا يناسب متعلمي المرحلة الثانوية، واعتماده على التكامل بين المفاهيم النظرية والتطبيقات العملية، ويُسهم في تنمية مهارات التفكير المنطقي والبرمجي لدى المتعلم، وتمكينه من توظيف قواعد البيانات بفاعلية في مشاريعه المستقبلية.

المؤلفون

الوحـدة الأولى- المُعالجة الرقمية
قواعد البيانات SQLite في Python

1. العلاقات في قواعد البيانات
   Relationships in Database
2. إنشاء الجداول المترابطة في قواعد البيانات
   Create Related Tables in Database
3. إضافة البيانات في الجداول المترابطة
   Inserting Data into Related Tables
4. الاستعلام عن البيانات من الجداول المترابطة
   Querying Data from Related Tables
5. تحديث البيانات في الجداول المترابطة
   Updating Data in Related Tables
6. حذف البيانات من الجداول المترابطة
   Deleting Data from Related Tables

العلاقات في قواعد البيانات
Relationships in Database

نتاجات التعلم

- تعريف مفهوم العلاقات بين الجداول وشرح دورها في تنظيم وربط البيانات في قواعد البيانات العلائقية.
- تمييز وظيفة كلٍّ من المفتاح الأساسي Primary Key والمفتاح الخارجي Foreign Key في ربط الجداول وضمان التكامل.
- تحديد دور كلٍّ من الجدول الأب Parent Table والجدول الابن Child Table عند إنشاء العلاقات.
- تفسير أهمية العلاقات في منع التكرار وضمان اتساق البيانات وتحسين جودة المعلومات.
- تمييز أنواع العلاقات الأساسية (واحد إلى واحد - واحد إلى متعدد - متعدد إلى متعدد)، وتحديد استخدام كل نوع.
- استخدام العلاقات في استخراج بيانات مترابطة من أكثر من جدول عبر الاستعلامات.
- تحليل فوائد العلاقات في تسهيل إدارة البيانات، تعديلها، وتحسين أداء النظام.
- تطبيق مبادئ تصميم الجداول باستخدام المُخططات ERD بما يُحسن هيكلة الجداول وتقليل تكرار البيانات داخل قاعدة البيانات.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم

العلاقات في قواعد البيانات العلائقية

تُعد العلاقات من أهم الركائز في بناء قواعد البيانات العلائقية التي تربط البيانات، وتضمن تنظيمها بطريقة منطقية تُسهل إدارتها Data Management، واسترجاعها Data Retrieval، وتحليلها Data Analysis، وإنشاء هياكل بيانات مُتماسكة Consistent Data Structures تمنع تكرار البيانات Data Redundancy، وتضمن سلامة واتساق البيانات Data Integrity.

تعريف العلاقات بين الجداول في قواعد البيانات

الروابط بين الجداول ذات البيانات المشتركة بهدف ربطها بطريقة منطقية ومنظّمة لإدارتها بكفاءة، وذلك بالاعتماد على المفاتيح بوصفها الأساس الذي تُبنى عليه العلاقات بين الجداول.

المفاتيح في العلاقات بين الجداول

تُستخدم المفاتيح Keys كجزء من القيود Constraints لضمان سلامة البيانات Data Integrity وتنظيم العلاقات بين الجداول، وهي:

المفتاح الأساسي Primary Key

قيد يُطبق على عمود أو مجموعة أعمدة في جدول ذات بيانات فريدة، وغير فارغة NOT NULL لتعريف كل صف بشكل فريد.

المفتاح الخارجي Foreign Key

قيد يُطبق على عمود أو أكثر في جدول، حيث تشير هذه الأعمدة إلى عمود أو أكثر (مفتاح أساسي أو فريد) في جدول آخر (أو في نفس الجدول)، وذلك لضمان صحة العلاقة وسلامة البيانات.

أنواع الجداول في العلاقات

في قواعد البيانات العلائقية، تختلف الجداول حسب دورها وطريقة استخدامها، ومنها:

الجدول الأب Parent Table

الجدول الذي يشير إليه قيد المفتاح الخارجي Foreign Key.

الجدول الابن Child Table

الجدول الذي يُطبق عليه قيد المفتاح الخارجي Foreign Key.

أهمية العلاقات في قواعد البيانات

تُساعد العلاقات بين الجداول على تنظيم البيانات وضمان صحتها وسهولة إدارتها، وتبرز أهميتها من خلال العناصر التالية:

ضمان التكامل Integrity

يُحافظ ضمان التكامل على صحة وترابط البيانات بين الجداول، ويمنع إدخال القيم غير الصحيحة، مما يضمن موثوقية المعلومات ودقة النتائج المستخرجة.

كفاءة التحديث Update Efficiency

تُتيح العلاقات سرعة تعديل البيانات دون التأثير على باقي النظام، مما يُسهل إدارة المعلومات وضمان استمرار عملها بكفاءة.

سهولة الاستعلام Querying

الوصول إلى المعلومات المطلوبة بسرعة ودقة، مما يُساعد على تحليل البيانات، واتخاذ القرارات بكفاءة.

ضمان الاتساق Consistency

الحفاظ على تناسق البيانات ودقتها عند تنفيذ العمليات المختلفة، مما يضمن أن تظل المعلومات موحّدة وصحيحة عبر جميع الجداول.

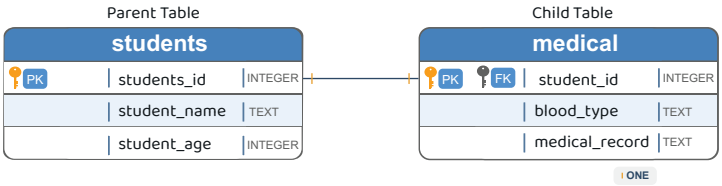
أنواع العلاقات في قواعد البيانات

تُصنف العلاقات في قواعد البيانات العلائقية إلى ثلاثة أنواع، وتُحدد نوع العلاقة بُناءً على طبيعة ارتباط البيانات بين الجداول.

علاقة واحد إلى واحد One to One

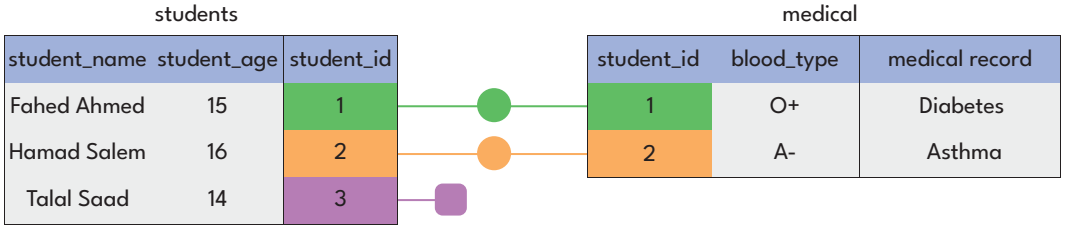
العلاقة التي يرتبط فيها كل صف في جدول الأب Parent بحد أقصى بصف واحد فقط في جدول الابن Child، بينما يجب أن يرتبط كل صف في الجدول الابن Child بصف واحد فقط في الجدول الأب Parent.

ويُمكن أن يكون العمود مفتاحًا أساسيًا ومفتاحًا خارجيًا في الوقت ذاته لفرض علاقة صارمة بين جدولين وضمان تطابق السجلات دون تكرار.



شكل (1-1) يُمثل العلاقة من نوع واحد إلى واحد.

يُوضح الشكل التالي أنه قد يكون لكل طالب سجل طبي واحد فقط أو لا يكون له سجل طبي، بينما كل سجل طبي مرتبط بطالب واحد فقط، بحيث لا يُسمح بوجود أكثر من ارتباط واحد بين الطالب والسجل الطبي.



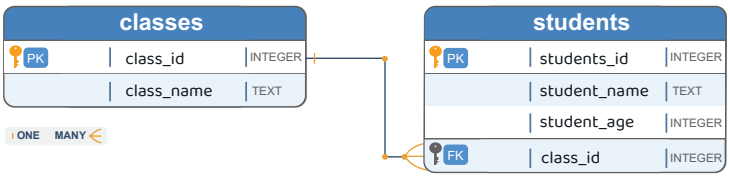
شكل (1-2) يُوضح بيانات جدولين يربطهما علاقة من نوع واحد إلى واحد.

لاحظ

الطالب رقم (3) في جدول student لا يملك سجلاً طبياً في جدول medical، إذ إن علاقة واحد إلى واحد لا تعني بالضرورة أن كل صف في الجدول الأب Parent يجب أن يقابله صف في الجدول الابن Child.

علاقة واحد إلى متعدد One to Many

العلاقة الأكثر شيوعًا في قواعد البيانات العلائقية، حيث كل صف في الجدول الأب Parent يرتبط بصف أو أكثر في الجدول الابن Child، وكل صف في الجدول الابن Child لا يرتبط بأكثر من صف واحد فقط في الجدول الأب Parent.



شكل (1-3) يُمثل العلاقة من نوع واحد إلى مُتعدد.

يُوضح الشكل التالي في العلاقة بين جدول الفصول classes، وجدول الطلاب students حيث كل فصل يحتوي على عدة طلاب، لكن كل طالب ينتمي إلى فصل واحد فقط.

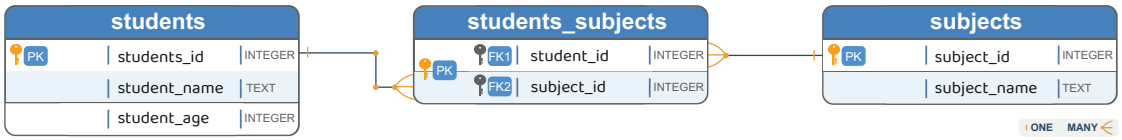
class_name	class_id
15	1
16	2
14	3

class_id	student_id	student_name	student_age
1	1	Fahed Ahmed	15
1	2	Hamad Salem	16
1	3	Talal Saad	14
2	4	Yossef Meshari	16
2	5	Mohamed Ali	14

شكل (1-4) يُوضح مثال عن بيانات جدولي يربطها العلاقة من نوع واحد إلى مُتعدد.

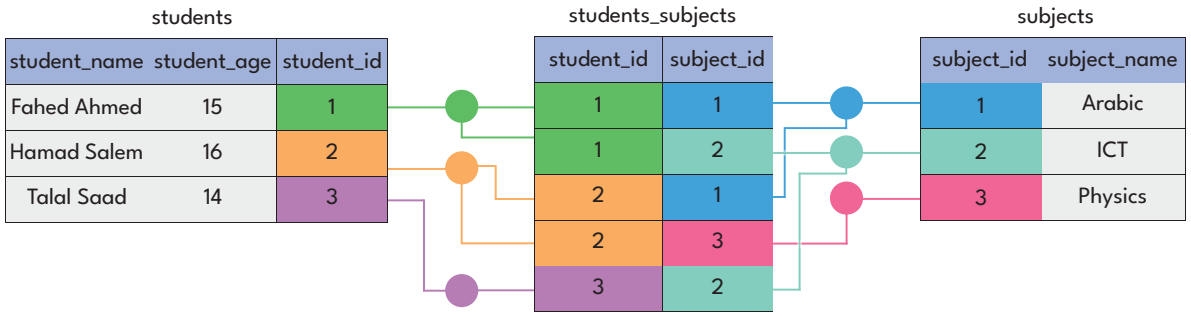
علاقة متعدد إلى متعدد Many to Many

العلاقة التي يُمكن أن يرتبط بها كل صف في جدول بعدة صفوف في الجدول الآخر، والعكس صحيح، ولا يمكن إنشاء هذه العلاقة بشكل مباشر دون استخدام جدول وسيط Junction Table.



شكل (1-5) يُمثل العلاقة من نوع متعدد إلى متعدد باستخدام جدول وسيط.

يُوضح الشكل التالي أن كل طالب في جدول students يدرس عدة مواد دراسية، والمادة الدراسية في جدول subjects يدرسها أكثر من طالب، وتم إنشاء جدول وسيط students_subjects لتمثيل العلاقة.



شكل (1-6) يُوضح مثال عن بيانات جداول يربطها العلاقة من نوع مُتعدد إلى مُتعدد.

يُساعد فهم أنواع العلاقات بين الجداول داخل قواعد البيانات في تمثيلها بصورة واضحة، الأمر الذي يُمهّد لتصميم مخطط الكيان والعلاقة (ERD) بوصفه الأداة التي تُجسِّد هذه العلاقات تمثيلاً رسوميًا منظّمًا.

مخططات الكيان والعلاقة Entity-Relationship Diagram (ERD)

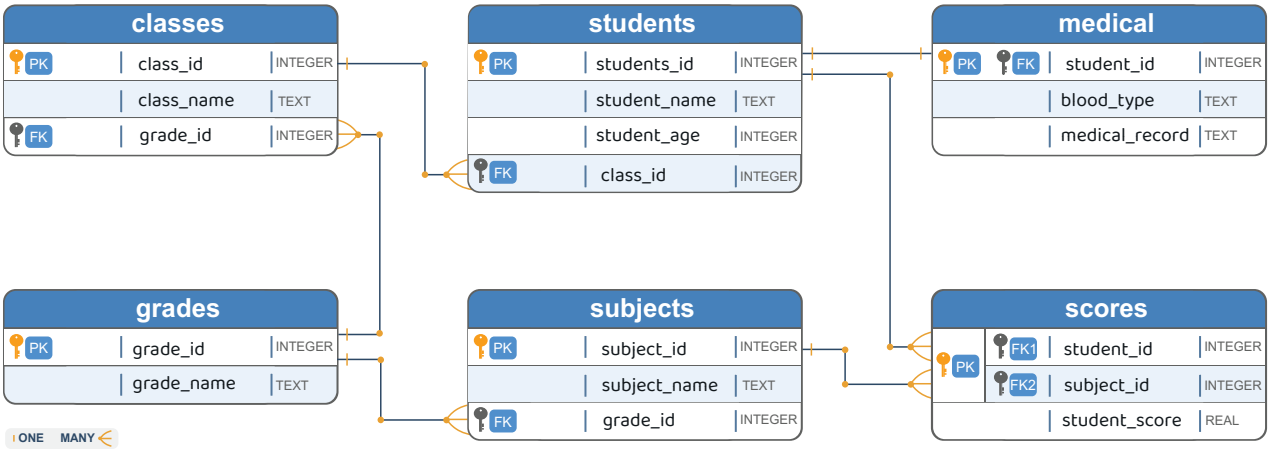
تمثيل رسومي لهياكل قواعد البيانات يوضّح الكيانات والعلاقات بينها. فعلى سبيل المثال، يُمثّل جدول الطلاب Entity، وتُمثّل الأعمدة مثل الاسم والعمر والصف Attributes تصف هذا الكيان، بينما يُمثّل كل صف في الجدول Instance لطالب واحد.

فوائد مخططات الكيان والعلاقة

تُسهم المخططات في تحسين تصميم هياكل قواعد البيانات من خلال:

1. التواصل بين فريق عمل المطورين.
2. تسهيل فهم البيانات وعلاقتها ببعضها.
3. اكتشاف الأخطاء المحتملة قبل بناء قاعدة البيانات.
4. تحسين تصميم الجداول، وتقليل التكرار.
5. العمل كوثيقة رسمية يتم الرجوع إليها أثناء التطوير.

مخطط الكيان والعلاقة ERD



شكل (1-7) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

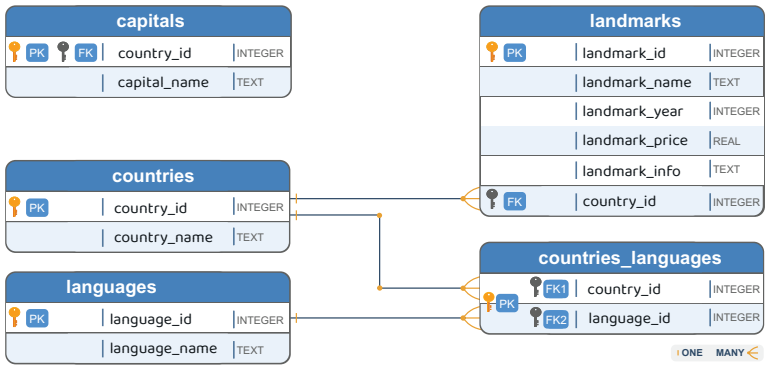
يُجسِّد المخطط قاعدة بيانات علائقية لإدارة بيانات الطلاب، حيث تشتمل على مجموعة من الجداول المترابطة فيما بينها. ويُراعى في هذا المخطط ما يلي:

- تمثيل الكيان على شكل مستطيل، يُكتب بداخله اسم الكائن، وخصائصه.
- التعبير عن المفتاح الأساسي Primary Key بالاختصار PK.
- التعبير عن المفتاح الخارجي Foreign Key بالاختصار FK.
- تمثيل العلاقة بين الكيانات بخطوط.
- توجد عدة طرق لتمثيل العلاقات، ومنها طريقة Crow's Foot Notation نستخدم منها:
  - علاقة واحد إلى واحد:
  - علاقة واحد إلى متعدد:
  - علاقة متعدد إلى متعدد:
- الجداول المترابطة بعلاقة واحد إلى مُتعدد:
  - الصفوف الدراسية grades، والفصول classes.
  - الصفوف الدراسية grades، والمواد الدراسية subjects.
  - الفصول classes، والطلاب students.
  - الطلاب students، والدرجات scores.
  - المواد الدراسية subjects، والدرجات scores.
- الجداول المترابطة بعلاقة واحد إلى واحد:
  - الطلاب students، والسجل الطبي medical.
- العلاقة بين جدول الطلاب students، وجدول المواد الدراسية subjects علاقة مُتعدد إلى مُتعدد.
- يُساهم التصميم الصحيح لقاعدة البيانات في منع تكرار البيانات، وضمان تكاملها بكفاءة.

أوراق العمل

ورقة عمل (1)　　التاريخ:

من خلال دراستك لمفاهيم مخطط الكيان والعلاقة (Entity Relationship Diagram -ERD)، ادرس مخطط الكيان والعلاقة لقاعدة البيانات countries_landmarks.db، مع الأخذ في الاعتبار أن الدولة لها عاصمة واحدة، ويُمكن أن يكون لها أكثر من لغة رسمية، وقد تُستخدم العملة الواحدة في أكثر من دولة، مما يستلزم تمثيل هذا الارتباط بطريقة مناسبة عند تصميم قاعدة البيانات:



شكل (1-8) يُمثل مخطط الكيان والعلاقة Entity Relationship Diagram لقاعدة البيانات countries_landmarks.

- حدد جدولي الأب والابن ونوع العلاقة بين جدولي landmarks وcountries:
  - الجدول الأب Parent: .....
  - الجدول الابن Child: .....
  - نوع العلاقة: .....
- اذكر نوع العلاقة واسم الجدول الوسيط بين الجدولين languages وcountries.
  - نوع العلاقة: .....
  - اسم الجدول الوسيط: .....
- ارسم العلاقة في المخطط بين جدولي countries وcapitals.

إنشاء الجداول المترابطة في قواعد البيانات
Create Related Tables in Database

نتاجات التعلم

- تعريف مفهوم علاقة واحد إلى متعدد One to Many ودورها في ربط الجداول وتنظيم البيانات.
- تمييز الجداول من حيث دورها في العلاقة: الجدول الأب Parent Table والجدول الابن Child Table.
- تحديد هيكلية الجدولين من حيث الأعمدة وأنواع البيانات والقيود المطلوبة لكل عمود.
- إنشاء الجدول الأب Parent والجدول الابن Child باستخدام أوامر CREATE TABLE في SQLite.
- تطبيق قيود المفاتيح الأساسية Primary Key والمفاتيح الخارجية Foreign Key لربط الجداول منطقياً.
- تنفيذ خيارات التكامل المرجعي مثل ON DELETE CASCADE وON UPDATE CASCADE لضمان اتساق البيانات.
- برمجة العلاقة One to Many في Python باستخدام حزمة sqlite3 وكتابة التعليمات البرمجية اللازمة.
- معاينة هيكلية الجداول والعلاقة بينهما في برنامج DB Browser for SQLite للتأكد من نجاح الربط.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم

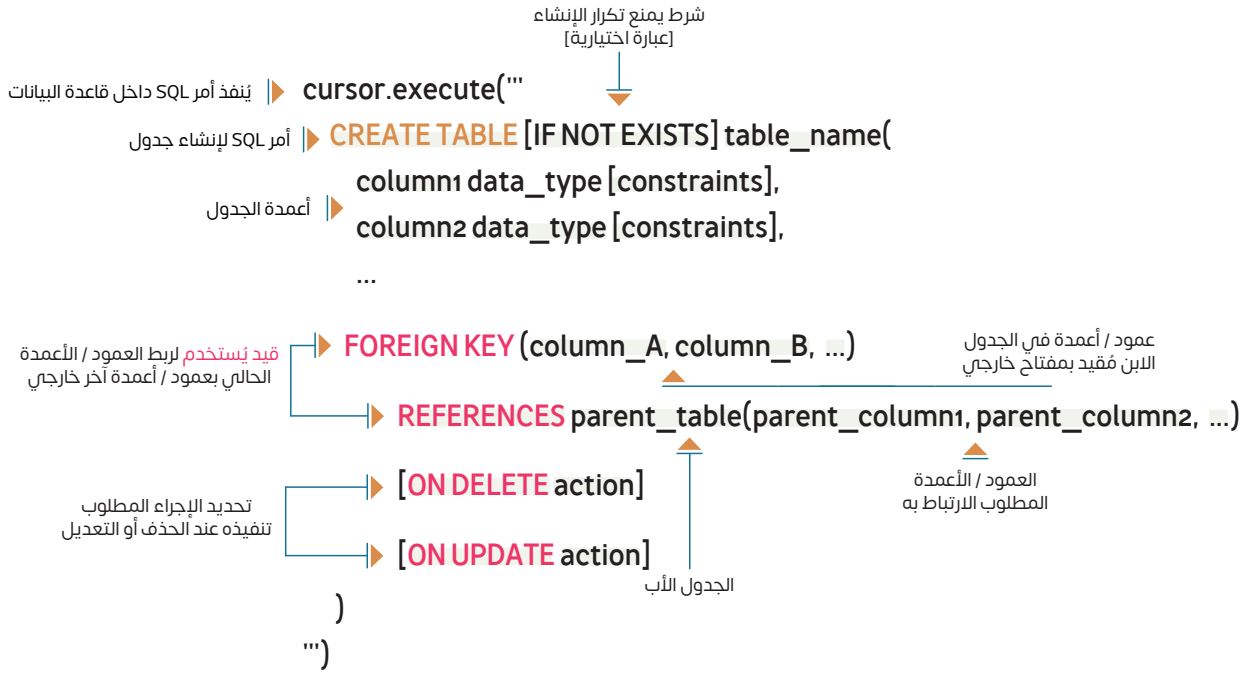
إنشاء الجداول في قواعد البيانات العلائقية

تُمثل العلاقات في قواعد البيانات الأساس الذي تُبنى عليه التطبيقات الحديثة لضمان دقة البيانات وترابطها ومرونتها بطريقة منظمة، وتُعد علاقة واحد إلى متعدد One to Many خطوة أساسية في تصميم قواعد البيانات.

يتم ذلك بإضافة مجموعة من القيود Constraints عند إنشاء الجداول أو تعديلها لضمان صحة ودقة البيانات، وتستخدم لمنع إدخال بيانات غير صحيحة أو غير مسموح بها.

صيغة التعليمة البرمجية لإنشاء جدول الابن Child في قاعدة البيانات SQLite في Python

يُمكن إنشاء جدول مع تحديد قيد Foreign Key بقاعدة البيانات SQLite في Python، مع تحديد الإجراء المطلوب عند حذف أو تعديل الصفوف في الجدول الأب Parent من خلال صيغة التعليمة البرمجية التالية:

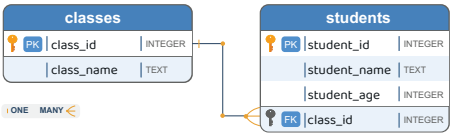


شكل (2-1) يُمثل صيغة التعليمة البرمجية لإنشاء جدول يتضمن مفتاح خارجي.

إنشاء جدول الأب Parent، وجدول الابن Child في قاعدة البيانات

مثال إنشاء الجدولين: جدول الفصول classes، وجدول الطلاب students في قاعدة البيانات school_data.db.

مطلوب: إنشاء جدولين مترابطين بعلاقة من نوع One to Many في قاعدة البيانات.



شكل (2-2) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

لاحظ

العلاقة السابقة من نوع واحد إلى متعدد One to Many حيث:

- classes هو الجدول الأب Parent.
- students هو الجدول الابن Child.
- class_id هو المفتاح المشترك بين الجدولين حيث يُمثل المُفتاح الأساسي في جدول classes، ويمثل المفتاح الخارجي في جدول students.

البيانات التالية توضح الأعمدة وأنواعها والقيود المطلوب تطبيقها على الجدولين:

1. جدول classes

اسم العمود	النوع	القيود
class_id	INTEGER	مفتاح أساسي PRIMARY KEY
class_name	TEXT	غير فارغ NOT NULL
		فريد UNIQUE

جدول (2-1) يُمثل جدول classes.

لاحظ

- يمنع القيد UNIQUE وجود فصلين بنفس الاسم في العمود class_name.

2. الجدول students

اسم العمود	النـــــوع	القيـــــود
student_id	INTEGER	مفتاح أساسي PRIMARY KEY
student_name	TEXT	غير فارغ NOT NULL
student_age	INTEGER	غير فارغ NOT NULL
class_id	INTEGER	غير فارغ NOT NULL
		مفتاح خارجي FOREIGN KEY يرتبط بالعمود class_id في الجدول classes

جدول (2-2) يُمثل جدول students.

يُستخدم القيد NOT NULL مع العمود class_id، لضمان عدم السماح بإدخال بيانات **طالب** دون تحديد **فصل** تابع له، بحيث تشير كل قيمة إلى فصل موجود في جدول classes، مما يحافظ على سلامة العلاقة بين الجدولين.

خطوات التنفيذ

- استيراد حزمة sqlite3.
- إنشاء / فتح اتصال بقاعدة البيانات.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- إنشاء الجدول الأب Parent.
- إنشاء الجدول الابن Child.
- حفظ التغيرات.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة هيكلية الجدول في برنامج DB Browser for SQLite.

```
import sqlite3
connection = sqlite3.connect('school_data.db')
cursor = connection.cursor()

cursor.execute("""
 CREATE TABLE IF NOT EXISTS classes (
  class_id      INTEGER   PRIMARY KEY,
  class_name    TEXT      NOT NULL      UNIQUE
 )
""")

cursor.execute("""
 CREATE TABLE IF NOT EXISTS students (
  student_id      INTEGER    PRIMARY KEY,
  student_name    TEXT       NOT NULL,
  student_age     INTEGER    NOT NULL,
  class_id        INTEGER    NOT NULL,
  FOREIGN KEY (class_id )
          REFERENCES classes (class_id)
          ON DELETE CASCADE
          ON UPDATE CASCADE
 )
""")

connection.commit()
connection.close()
```

التفسيـــــر

```
import sqlite3
```

- استيراد حزمة sqlite3.

```
connection = sqlite3.connect('school_data.db')
```

- الاتصال بقاعدة البيانات school_data.db.

```
cursor = connection.cursor()
```

- إنشاء كائن المؤشر cursor.

```
cursor.execute('''
    CREATE TABLE IF NOT EXISTS classes (
        class_id        INTEGER     PRIMARY KEY,
        class_name      TEXT        NOT NULL        UNIQUE
    )
''')
```

- إنشاء الجدول classes (الجدول الأب Parent Table).

```
cursor.execute('''
    CREATE TABLE IF NOT EXISTS students (
        student_id      INTEGER     PRIMARY KEY,
        student_name    TEXT        NOT NULL,
        student_age     INTEGER     NOT NULL,
        class_id        INTEGER     NOT NULL,
        FOREIGN KEY (class_id)
            REFERENCES classes(class_id)
            ON UPDATE CASCADE
            ON DELETE CASCADE
    )
''')
```

- إنشاء الجدول students (الجدول الابن Child Table).

لاحـــظ

يتم إنشاء القيد Foreign key بالصيغة التالية:

```
FOREIGN KEY (class_id)
        REFERENCES classes (class_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
```

- class_id: العمود المطلوب إنشاء القيد عليه (Foreign key).
- classes: جدول الأب Parent.
- class_id: العمود المطلوب الارتباط به في جدول الأب Parent.
- ON DELETE CASCADE: عند حذف صف في جدول الأب Parent، تُحذف كل الصفوف المرتبطة في جدول الابن Child تلقائياً.
- ON UPDATE CASCADE: عند تعديل قيمة المفتاح الأساسي في جدول الأب Parent، يتم تحديث قيم المفتاح الخارجي المرتبطة في جدول الابن Child تلقائياً.

```
connection.commit()
```

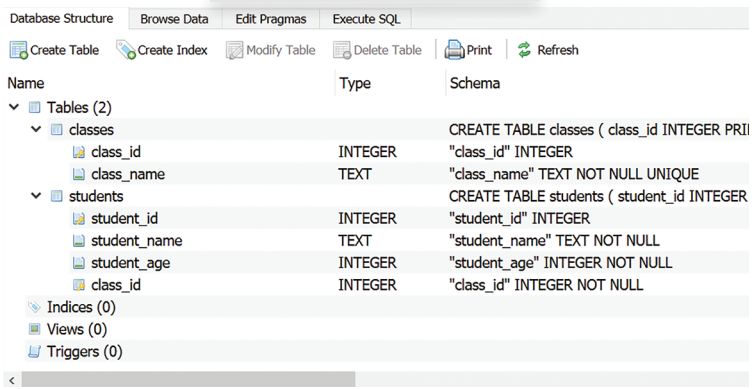
- حفظ التغييرات.

```
connection.close()
```

- إغلاق الاتصال بقاعدة البيانات.

معاينة هيكلية قاعدة البيانات في برنامج DB Browser for SQLite

بعد إنشاء الجداول



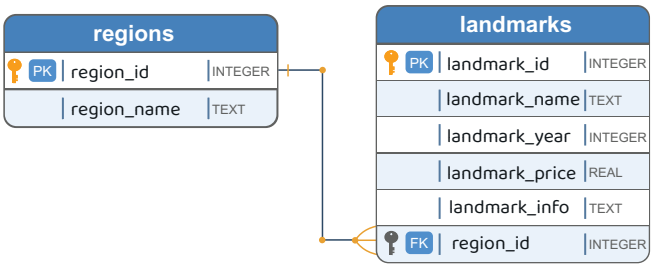
شكل (2-3) يُوضح هيكلية قاعدة بيانات school_data.db.

أوراق العمل

ورقة عمل (2)　　التاريخ:

إنشاء جدولين بقاعدة البيانات Kuwait_landmarks.db:

- **الجدول regions: لتخزين مناطق معالم دولة الكويت.**
- **الجدول landmarks: لتخزين بيانات بعض معالم دولة الكويت.**



شكل (2-4) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

1. استدعِ المشروع landmark_create.
2. افتح ملف create_table.py.

```
1   import sqlite3
2   connection = sqlite3.connect('Kuwait_landmarks.db')
3   cursor = connection.cursor()
4
5   cursor.execute('''
6           CREATE TABLE IF NOT EXISTS regions (
7                   region_id       INTEGER     .....,
8                   region_name     TEXT        NOT NULL  UNIQUE
9           )
10  ''')
11
```

```
12  cursor.execute('''
13          CREATE TABLE IF NOT EXISTS landmarks (
14                  landmark_id       INTEGER     PRIMARY KEY,
15                  landmark_name     TEXT        NOT NULL      UNIQUE,
16                  landmark_year     INTEGER,
17                  landmark_price    REAL        DEFAULT 0,
18                  landmark_info     TEXT,
19                  region_id         INTEGER     NOT NULL,
20
21                  FOREIGN KEY (.....)
22                      REFERENCES regions (.....)
23                          ON UPDATE CASCADE
24                          ON DELETE CASCADE
25          )
26  ''')
27
28  connection.commit()
29  connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:

◄ أنشئ جدول regions، وفقاً للبيانات التالية:

اسم العمود	النوع	القيود
region_id	INTEGER	المفتاح الأساسي PRIMARY KEY
region_name	TEXT	غير فارغ NOT NULL
		فريد UNIQUE

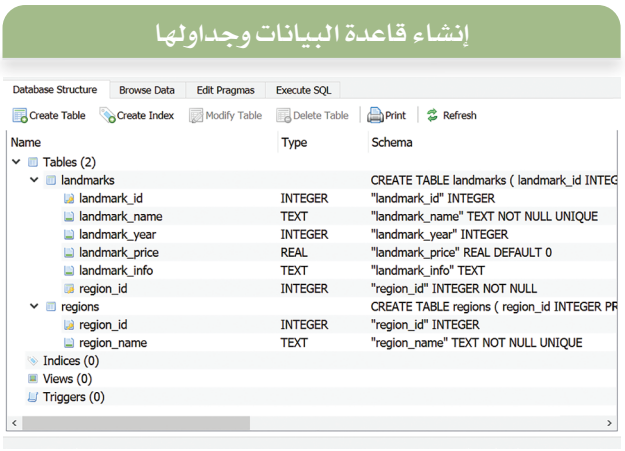
جدول (2-3) يُمثل جدول regions.

◄ أنشئ جدول landmarks، وفقاً للبيانات التالية:

اسم العمود	النوع	القيود
landmark_id	INTEGER	المفتاح الأساسي PRIMARY KEY
landmark_name	TEXT	غير فارغ NOT NULL
		فريد UNIQUE
landmark_year	INTEGER	-
landmark_price	REAL	القيمة الافتراضية صفر DEFAULT 0
landmark_info	TEXT	-
region_id	INTEGER	غير فارغ NOT NULL
		FOREIGN KEY (region_id) REFERENCES regions(region_id) ON UPDATE CASCADE ON DELETE CASCADE

جدول (2-4) يُمثل جدول landmarks.

4. نفذ التعليمات البرمجية.
5. عاين هيكلية الجداول في برنامج DB Browser for SQLite.



شكل (2-5) يُوضح هيكلية قاعدة البيانات Kuwait_landmarks.db.

إضافة البيانات في الجداول المترابطة
Inserting Data into Related Tables

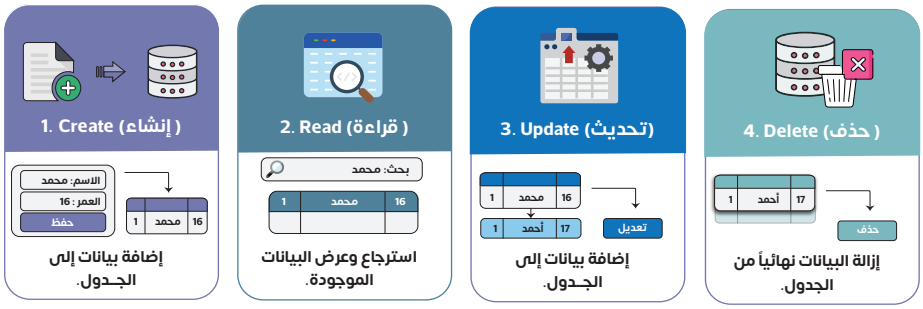
نتاجات التعلم

- تعريف عمليات CRUD الأساسية (Create, Read, Update, Delete)، ودورها في إدارة البيانات داخل قواعد البيانات.
- تفسير كيفية إدراج بيانات في جدولين مرتبطين بعلاقة One to Many أو One to One وفق شروط التكامل المرجعي.
- تحديد القيم المطلوبة لإضافة سجل جديد في جدول الأب Parent.
- استخدام قيمة المفتاح الأساسي للجدول الأب Parent لإدراج السجل المقابل له في جدول الابن Child بطريقة صحيحة.
- تنفيذ التعليمة INSERT في SQLite لإضافة بيانات جديدة إلى الجداول مع الالتزام بالقيود المطبقة.
- برمجة عملية الإدراج باستخدام Python وحزمة sqlite3، مع تفعيل دعم المفاتيح الخارجية لمنع الأخطاء.
- استخدام الاستعلام الفرعي Subquery لجلب البيانات من الجدول.
- معاينة البيانات المدخلة والتحقق من نجاح عملية الإدراج باستخدام DB Browser for SQLite.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم

التعامل مع قواعد البيانات العلائقية

يوجد أربع عمليات أساسية في قواعد البيانات، وتشمل إنشاء وإدخال البيانات Create، وقراءتها Read، وتحديثها Update، وحذفها Delete، وهي الأساس في التعامل مع أي نظام لإدارة قواعد البيانات DBMS.



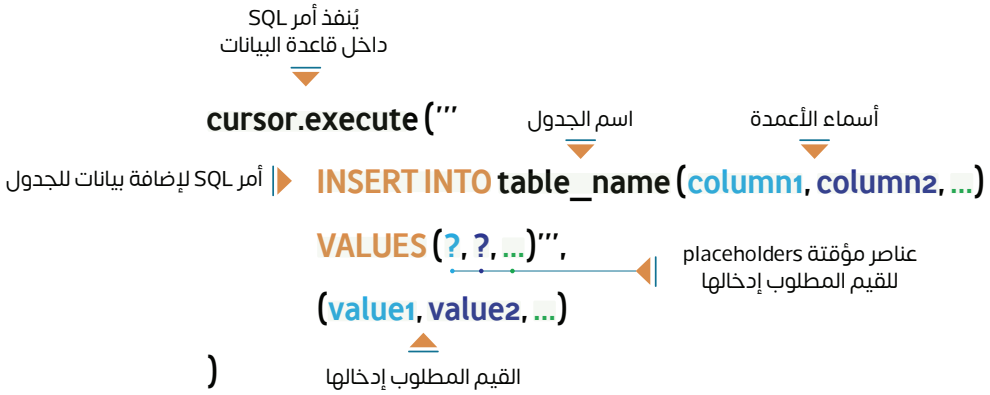
شكل (3-1) يُمثل العمليات الأساسية في قواعد البيانات.

إضافة البيانات في قواعد البيانات العلائقية

عند إضافة بيانات في جدولين مرتبطين بعلاقة واحد إلى متعددOne to Many، أو علاقة واحد إلى واحد One to One يجب مراعاة الترتيب التالي:

1. التأكد من وجود الصف في جدول الأب Parent أو إضافته في حال عدم وجوده.
2. تحديد قيمة المفتاح الأساسي للصف الموجود في جدول الأب Parent.
3. استخدام قيمة المفتاح الأساسي المحدد لإضافة السجل المرتبط به في جدول الابن Child.

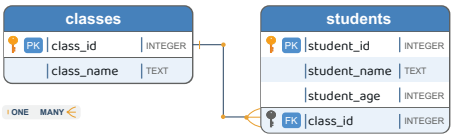
صيغة التعليمة البرمجية لإضافة البيانات SQLite في Python



شكل (3-2) يُمثل صيغة كتابة التعليمة البرمجية لإضافة البيانات.

مثال 1

إضافة بيانات في جدول classes بقاعدة البيانات school_data.db.



شكل (3-3) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

المطلوب: إدخال البيانات التالية في الجدول classes:

class_id	class_name
105	11S3

جدول (3-1) يُمثل بيانات الصف المطلوب إضافته للجدول.

خطوات التنفيذ

- استيراد حزمة sqlite3.
- استقبال قيم أعمدة جدول الأب Parent.
- إنشاء / فتح اتصال بقاعدة البيانات.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- إضافة البيانات إلى جدول الأب Parent.
- طباعة عدد الصفوف التي تمّت إضافتها.
- حفظ التغييرات.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة البيانات في برنامج DB Browser for SQLite.

```
1   import sqlite3
2
3   var_class_id = int(input('Enter class id: '))          # 105
4   var_class_name = input('Enter class name: ')           # 11S3
5
6   connection = sqlite3.connect('school_data.db')
7   cursor = connection.cursor()
8
9   cursor.execute('''
10          INSERT INTO classes (class_id,class_name)
11          VALUES(?,?)''',
12          (var_class_id,var_class_name)
13  )
14  print('Number of rows inserted= ', connection.total_changes)
15
16  connection.commit()
17  connection.close()
```

التفسيـــــر

```
import sqlite3
```

- استيراد حزمة sqlite3.

```
var_class_id = int(input('Enter class id: '))          # 105
var_class_name = input('Enter class name: ')           # 11S3
```

- استقبال من المستخدم القيم المطلوبة.

```
connection = sqlite3.connect('school_data.db')
```

- الاتصال بقاعدة بيانات school_data.db.

```
cursor = connection.cursor()
```

- إنشاء كائن المؤشر cursor.

```
cursor.execute('''
        INSERT INTO classes (class_id, class_name)
        VALUES(?, ?)''',
        (var_class_id, var_class_name)
)
```

- إضافة البيانات للجدول classes بطريقة آمنة، مع ضمان عدم تكرار القيم في عمود class_name نظرًا لتطبيق القيد unique عليه عند إنشاء الجدول.

```
print('Number of rows inserted = ', connection.total_changes)
```

- طباعة إجمالي عدد الصفوف التي حدث لها تغيير في قاعدة البيانات.

```
connection.commit()
```

- حفظ التغييرات.

```
connection.close()
```

- إغلاق الاتصال بقاعدة البيانات.

Program Output

```
Enter class id: 105
Enter class name: 11S3
Number of rows inserted = 1
```

معاينة بيانات الجداول في برنامج DB Browser for SQLite

قبل إضافة البيانات

Table: classes

	class_id	class_name
	Filter	Filter
1	101	11S1
2	102	11S2
3	103	11A1
4	104	11A2

بعد إضافة البيانات

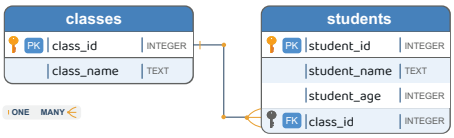
Table: classes

	class_id	class_name
	Filter	Filter
1	101	11S1
2	102	11S2
3	103	11A1
4	104	11A2
5	105	11S3

شكل (3-4) يُوضح بيانات جدول classes قبل وبعد إضافة صف جديد.

مثال 2

إضافة البيانات التالية في جدول students بقاعدة البيانات school_data.db



شكل (3-5) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

المطلوب: إدخال البيانات التالية في الجدول students:

student_id	student_name	student_age	class_id
	Ahmed Ali Saad	16	103

جدول (3-2) يُمثل بيانات الصف المطلوب إضافته للجدول.

خطوات التنفيذ

- استيراد حزمة sqlite3.
- استقبال قيم أعمدة جدول الابن Child.
- إنشاء/ فتح اتصال بقاعدة البيانات.
- تفعيل دعم المفاتيح الخارجية في قاعدة بيانات.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- إضافة البيانات في جدول الابن Child.
- طباعة عدد الصفوف التي تمّت إضافتها.
- حفظ التغييرات.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة البيانات في برنامج DB Browser for SQLite.

```
1   import sqlite3
2
3   var_student_name = input('Enter student name: ')  # Ahmed Ali Saad
4   var_student_age = int(input('Enter student age: '))  # 16
5   var_class_id = int(input('Enter class  id: '))          # 103
6
7   connection = sqlite3.connect('school_data.db')
8   connection.execute('PRAGMA foreign_keys = ON')
9   cursor = connection.cursor()
10
11  cursor.execute('''
12      INSERT INTO students (student_name, student_age, class_id)
13      VALUES(?, ?, ?)''',
14      (var_student_name, var_student_age, var_class_id)
15  )
16  print('Number of rows inserted= ', connection.total_changes)
17
18  connection.commit()
19  connection.close()
```

التفسيـــــر

- استيراد الحزمة sqlite3.

```
import sqlite3
```

```
var_student_name =input('Enter student name: ')  # Ahmed Ali Saad
var_student_age = int(input('Enter student age: '))  # 16
var_class_id = int(input('Enter class  id: '))          # 103
```

- استقبال من المستخدم القيم المطلوبة.

```
connection = sqlite3.connect('school_data.db')
```

- الاتصال بقاعدة بيانات school_data.db.

```
connection.execute('PRAGMA foreign_keys = ON ')
```

- تفعيل دعم المفاتيح الخارجية.

تستخدم العبارة لمنع إدخال صف في جدول الابن Child غير مرتبط بجدول الأب Parent، وبذلك يضمن سلامة العلاقات المرجعية بين الجداول.

```
cursor = connection.cursor()
```

- إنشاء كائن المؤشر cursor.

```
cursor.execute('''
        INSERT INTO students (student_name, student_age, class_id)
        VALUES(?, ?, ?)''',
        (var_student_name, var_student_age, var_class_id)
)
```

- إضافة بيانات طالب جديد للفصل.

```
print('Number of rows inserted= ', connection.total_changes)
```

- طباعة إجمالي عدد الصفوف التي حدث لها تغيير في قاعدة البيانات.

```
connection.commit()
```

- حفظ التغييرات.

```
connection.close()
```

- إغلاق الاتصال بقاعدة البيانات.

Program Output

Enter student name: Ahmed Ali Saad ↵ Enter
Enter student age: 16 ↵ Enter
Enter class id: 103 ↵ Enter
Number of rows inserted = 1

معاينة بيانات الجداول في برنامج DB Browser for SQLite

قبل إضافة البيانات

Table: students

	student_id	student_name	student_age	class_id
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	102
4	4	Saud Abdullah Yousef	15	102
5	5	Fahad Farhan Ahmed	15	103
6	6	Abdullateef Ali Hussain	16	103

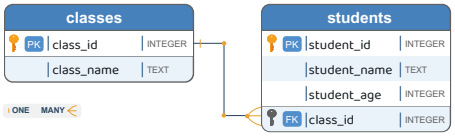
بعد إضافة البيانات

Table: students

	student_id	student_name	student_age	class_id
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	102
4	4	Saud Abdullah Yousef	15	102
5	5	Fahad Farhan Ahmed	15	103
6	6	Abdullateef Ali Hussain	16	103
7	7	Ahmed Ali Saad	16	103

شكل (3-6) يُوضح بيانات جدول students قبل وبعد إضافة صف جديد.

مثال 3 إضافة البيانات التالية في جدول students باستخدام (الاستعلام الفرعي) بقاعدة البيانات school_data.db:



شكل (3-7) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

المطلوب: إدخال البيانات التالية في الجدول students:

جدول classes

class_id	class_Name
103	11S3

جدول students

student_id	student_name	student_age	class_id
	Bader Abdullah Yousef	15	

شكل (3-8) يوضح بيانات الصف المطلوب إدخاله لجدول الابن Child بالاستعانة ببيانات جدول الأب Parent.

خطوات التنفيذ

- استيراد حزمة sqlite3.
- استقبال قيم أعمدة جدول الابن Child.
- إنشاء/ فتح اتصال بقاعدة البيانات.
- تفعيل دعم المفاتيح الخارجية في قاعدة بيانات.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- إضافة البيانات في جدول الابن Child مستعيناً بجدول الأب Parent.
- طباعة عدد الصفوف التي تمّت إضافتها.
- حفظ التغييرات.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة البيانات في برنامج DB Browser for SQLite.

```
1   import sqlite3
2
3   var_student_name = input('Enter student name: ')    # Bader Abdullah Yousef
4   var_student_age = int(input('Enter student age: '))  # 15
5   var_class_name = input('Enter class  name: ')        # 11S3
6
7   connection = sqlite3.connect('school_data.db')
8   connection.execute('PRAGMA foreign_keys = ON')
9   cursor = connection.cursor()
10
11  cursor.execute('''
12      INSERT INTO students (student_name, student_age, class_id)
13      VALUES(?, ?, (SELECT class_id FROM classes WHERE class_name = ?))''',
14      (var_student_name, var_student_age, var_class_name)
15  )
16  print('Number of rows inserted= ', connection.total_changes)
17
18  connection.commit()
19  connection.close()
```

التفسيـــــر

```
import sqlite3
```

- استيراد الحزمة sqlite3.

```
var_student_name =input('Enter student name: ')      # Bader Abdullah Yousef
var_student_age = int(input('Enter student age: '))  # 15
var_class_name = input('Enter class  name: ')        # 11S3
```

- استقبال من المستخدم القيم المطلوبة.

```
connection = sqlite3.connect('school_data.db')
```

- الاتصال بقاعدة بيانات school_data.db.

```
connection.execute('PRAGMA foreign_keys = ON ')
```

- تفعيل دعم المفاتيح الخارجية.

تُستخدم العبارة لمنع إدخال صف في جدول الابن Child غير مرتبط بجدول الأب Parent، وبذلك يضمن سلامة العلاقات المرجعية بين الجداول.

```
cursor = connection.cursor()
```

إنشاء كائن المؤشر cursor.

```
cursor.execute('''
        INSERT INTO students (student_name, student_age, class_id)
        VALUES(?, ?, (SELECT class_id FROM classes WHERE class_name = ?))''',
        (var_student_name, var_student_age, var_class_name)
)
```

إضافة بيانات طالب جديد للفصل.

تم استخدام الاستعلام الفرعي:
(SELECT class_id FROM classes WHERE class_name = ?)
للحصول على رقم الفصل للطالب من جدول classes بمعلومية اسم الفصل.

```
print('Number of rows inserted= ', connection.total_changes)
```

طباعة إجمالي عدد الصفوف التي حدث لها تغيير في قاعدة البيانات.

```
connection.commit()
```

حفظ التغييرات.

```
connection.close()
```

إغلاق الاتصال بقاعدة البيانات.

Program Output

```
Enter student name: Bader Abdullah Yousef
Enter student age: 15
Enter class  name: 11S3
Number of rows inserted = 1
```

معاينة بيانات الجداول في برنامج DB Browser for SQLite

قبل إضافة البيانات

Table: students

	student_id	student_name	student_age	class_id
	Filter	Filter	Filter	Filter
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	102
4	4	Saud Abdullah Yousef	15	102
5	5	Fahad Farhan Ahmed	15	103
6	6	Abdullateef Ali Hussain	16	103
7	7	Ahmed Ali Saad	16	103

بعد إضافة البيانات

Table: students

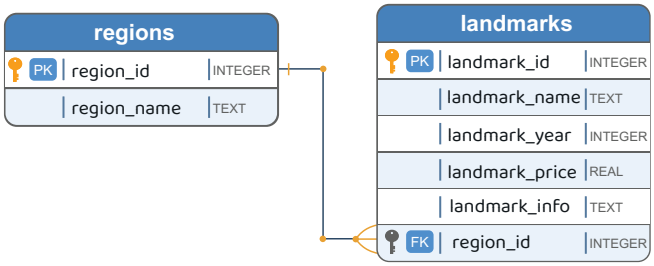
	student_id	student_name	student_age	class_id
	Filter	Filter	Filter	Filter
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	102
4	4	Saud Abdullah Yousef	15	102
5	5	Fahad Farhan Ahmed	15	103
6	6	Abdullateef Ali Hussain	16	103
7	7	Ahmed Ali Saad	16	103
8	8	Bader Abdullah Yousef	15	105

شكل (3-9) يُوضح بيانات جدول students قبل وبعد إضافة صف جديد.

أوراق العمل

ورقة عمل (3) التاريخ:

إضافة صف جديد لجدول landmarks بقاعدة البيانات Kuwait_landmarks.db.



شكل (3-10) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

1. استدعِ المشروع landmark_insert1.
2. افتح ملف data_insert.py

```
import sqlite3
# Get landmark data from user
var_landmark_name = input('Enter landmark name: ')    # Al-Tahrir Tower
var_landmark_year = int(input('Enter landmark year: ')) # 1996
var_region_id = int(input('Enter region id: '))        # 101

connection = sqlite3.connect('Kuwait_landmarks.db')
connection.execute('PRAGMA foreign_keys = ON')
cursor = connection.cursor()

# insert data into landmarks table
cursor.execute('''
        INSERT INTO landmarks
                (…… , …… , ……)
        VALUES (?,?,?) ''',
        (var_landmark_name, var_landmark_year, var_region_id)
)
print('Number of rows inserted= ', connection.total_changes)

```

```
20  connection.commit()
21  connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:

◄ أضف صف جديد في جدول landmarks، وفقاً للبيانات التالية:

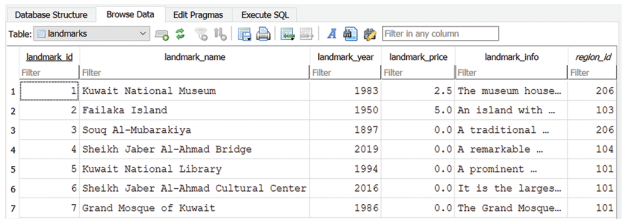
Column names	landmark_id	landmark_name	region_id	landmark_year	landmark_price	landmark_info
Data values		Al-Tahrir Tower	101	1996		

جدول (3-3) يُمثل بيانات الصف المطلوب إضافته للجدول.

4. نفذ التعليمات البرمجية.

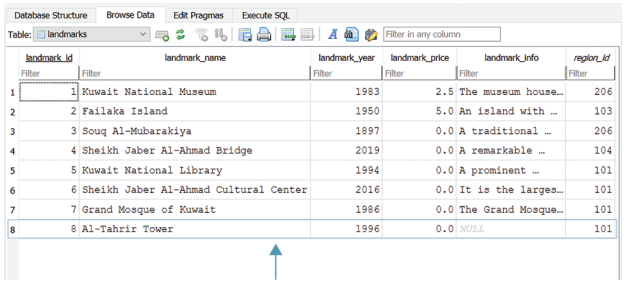
5. عاين بيانات الجداول في برنامج DB Browser for SQLite.

جدول landmarks قبل الإضافة



	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum house…	206
2	2	Failaka Island	1950	5.0	An island with …	103
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional …	206
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable …	104
5	5	Kuwait National Library	1994	0.0	A prominent …	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the larges…	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque…	101

جدول landmarks بعد الإضافة

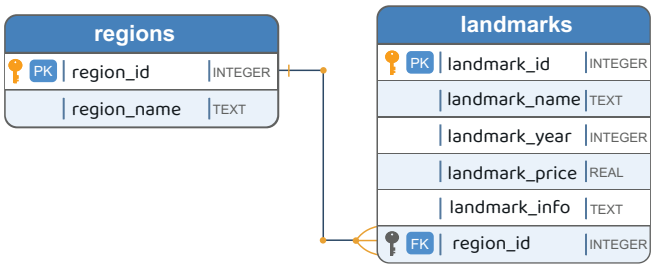


	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum house…	206
2	2	Failaka Island	1950	5.0	An island with …	103
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional …	206
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable …	104
5	5	Kuwait National Library	1994	0.0	A prominent …	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the larges…	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque…	101
8	8	Al-Tahrir Tower	1996	0.0	NULL	101

شكل (3-11) يُوضح بيانات جدول landmarks قبل وبعد الإضافة.

ورقة عمل (4) التاريخ:

إضافة صف جديد لجدول landmarks باستخدام الاستعلام الفرعي بقاعدة البيانات Kuwait_landmarks.db.



شكل (3-12) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

1. استدعِ المشروع landmark_insert2.
2. افتح مَلف data_insert.py.

```
import sqlite3
# Get landmark data from user
var_landmark_name = input('Enter landmark name: ') # Museum of Modern Art
var_landmark_year = int(input('Enter landmark year: ')) # 1980
var_region_name = input('Enter region name: ')          # Sharq

connection = sqlite3.connect('Kuwait_landmarks.db')
connection.execute('PRAGMA foreign_keys = ON')
cursor = connection.cursor()

# insert data into landmarks table
cursor.execute('''
        INSERT INTO landmarks
                (landmark_name, landmark_year, region_id)
        VALUES (?,?,
                (SELECT …….. FROM regions WHERE …….. =?)
        ) ''',
        (var_landmark_name, var_landmark_year, var_region_name)
)
print('Number of rows inserted= ', connection.total_changes)

connection.commit()
connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:

◄ أضف صفًا جديدًا في جدول landmarks، وفقاً للبيانات التالية:

Column names	landmark_id	landmark_name	region_id	landmark_year	landmark_price	landmark_info
Data values		Museum of Modern Art		1980		

جدول landmarks

Column names	region_id	region_name
Data values	105	Sharq

جدول regions

شكل (3-13) يوضح بيانات الصف المطلوب إدخاله لجدول الابن Child بالاستعانة ببيانات جدول الأب Parent.

4. نفذ التعليمات البرمجية.

5. عاين بيانات الجداول في برنامج DB Browser for SQLite.

جدول landmarks قبل الإضافة

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: landmarks

	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum houses…	206
2	2	Failaka Island	1950	5.0	An island with …	103
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional …	206
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable …	104
5	5	Kuwait National Library	1994	0.0	A prominent …	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the largest…	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque …	101
8	8	Al-Tahrir Tower	1996	0.0	NULL	101

جدول landmarks بعد الإضافة

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: landmarks

	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum houses…	206
2	2	Failaka Island	1950	5.0	An island with …	103
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional …	206
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable …	104
5	5	Kuwait National Library	1994	0.0	A prominent …	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the largest…	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque …	101
8	8	Al-Tahrir Tower	1996	0.0	NULL	101
9	9	Museum of Modern Art	1980	0.0	NULL	105

شكل (3-14) يُوضح بيانات جدول landmarks قبل وبعد الإضافة.

الاستعلام عن البيانات من الجداول المترابطة
Querying Data from Related Tables

نتاجات التعلم

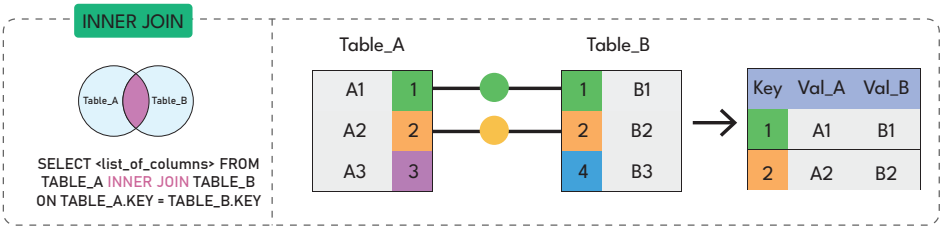
- تعريف مفهوم الاستعلام Query وبيان دوره في استرجاع البيانات من قاعدة البيانات بدقة.
- تمييز الأنواع الأساسية لعمليات الربط JOIN Types، وتحديد استخدام كل منهما.
- تفسير كيفية استخدام INNER JOIN لدمج البيانات من جدولين مرتبطين وعرض نتائج مشتركة ضمن استعلام واحد.
- اختيار الأعمدة المطلوبة من الجداول المختلفة عند تنفيذ الاستعلام، مثل student_name وclass_name.
- تنفيذ استعلامات INNER JOIN في SQLite باستخدام Python لاسترجاع بيانات مترابطة بناءً على شرط محدد.
- استخدام الدوال التجميعية مثل COUNT لحساب عدد السجلات وفق شروط محددة داخل الاستعلام.
- برمجة استعلامات SELECT مع WHERE وJOIN بشكل آمن باستخدام معاملات Placeholders في Python.
- معاينة نتائج الاستعلام وفحصها عبر طباعة المخرجات أو من خلال DB Browser for SQLite للتحقق من صحتها.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم

الاستعلام عن البيانات في قواعد البيانات العلائقية Querying Data

يُعدّ الاستعلام عن البيانات في قواعد البيانات العلائقية (Querying Data) من العمليات الأساسية التي تُمكّن المستخدم من الوصول إلى البيانات المخزّنة واستخراجها وفق شروط ومعايير محددة لعرض الصفوف المشتركة بين الجداول المترابطة.

تُستخدم عبارات JOIN لدمج بيانات أكثر من جدول، ثم ترتيب النتائج وتصفيتها، بما يدعم تحليل البيانات واتخاذ القرارات داخل التطبيقات المختلفة، ويُعد INNER JOIN من أبرزها.



شكل (4-1) يُمثل نوع الربط INNER JOIN.

صيغة التعليمة البرمجية لإنشاء استعلامات SQLite باستخدام INNER JOIN في Python

```
cursor.execute('''
    SELECT table1.column1, table1.column2,
           table2.column1, table2.column2,
           ...
    FROM table1
    INNER JOIN table2
        ON table1.columnX = table2.columnY
    WHERE condition_column = ?''',
    (condition_val,)
)
```

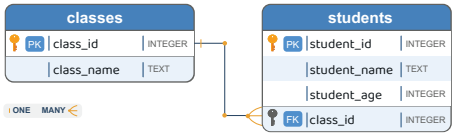
- `cursor.execute('''` : يُنفذ أمر SQL داخل قاعدة البيانات
- `SELECT ...` : أمر SQL لإسترجاع البيانات
- أسماء الأعمدة المطلوب استرجاعها
- `INNER JOIN table2` : عملية الربط يمكن استبداله بنوع آخر
- `ON table1.columnX = table2.columnY` : الأعمدة المستخدمة في الربط بين الجدولين قد يكون أحدهما مفتاحا أساسيا Primary Key والآخر مفتاحا خارجيا Foreign Key
- `WHERE condition_column = ?''',` : عُنصر مُؤقت placeholder لقيمة الشرط
- `(condition_val,)` : قيمة الشرط

شكل (4-2) يُمثل صيغة كتابة التعليمة البرمجية لإنشاء استعلامات باستخدام INNER JOIN.

الاستعلام عن بيانات وفق شرط محدد

مثال 1

الاستعلام عن بيانات طلاب محددة من قاعدة البيانات school_data.db.



شكل (3-4) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

المطلوب: كتابة التعليمة البرمجية لتنفيذ التالي:

- الاستعلام عن بيانات الطلاب (student_name, student_age, class_name) من الجدولين (students - classes).
- بناءً على الشرط (class_name = '11S1').
- ترتيب النتائج تصاعديًا وفقًا لأسماء الطلاب (student_name).

خطوات التنفيذ

- استيراد حزمة sqlite3.
- استقبال الفصل المراد البحث عنه.
- إنشاء / فتح اتصال بقاعدة البيانات.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- استرجاع البيانات.
- طباعة النتائج.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة النتائج.

```
import sqlite3

var_class_name = input('Enter class name: ')      # 11S1
connection = sqlite3.connect('school_data.db')
cursor = connection.cursor()

cursor.execute('''
    SELECT student_name, student_age
    FROM students
    INNER JOIN classes
        ON students.class_id = classes.class_id
    WHERE class_name = ?
    ORDER BY student_name ''',
    (var_class_name,)
)
rows=cursor.fetchall()
for row in rows:
    print(row)

connection.close()
```

التفسيـــــر

```
import sqlite3
```

- استيراد الحزمة sqlite3.

```
var_class_name = input('Enter class name: ')    # 11S1
```

- استقبال اسم الفصل، وتخزينه في المتغير var_class_name.

```
connection = sqlite3.connect('school_data.db')
```

- الاتصال بقاعدة بيانات school_data.db.

```
cursor = connection.cursor()
```

- إنشاء كائن المؤشر cursor.

```
cursor.execute('''
        SELECT student_name, student_age
        FROM students
        INNER JOIN classes
                ON students.class_id = classes.class_id
        WHERE class_name = ?
        ORDER BY student_name ''',
        (var_class_name,)
)
```

استخدام:

- عبارة SELECT لتحديد الأعمدة المطلوب عرضها.
- عبارة INNER JOIN لدمج الجدولين students وclasses بناءً على عمود الربط.
- عبارة WHERE لتصفية النتائج بناءً على الشرط المطلوب.
- عبارة ORDER BY لترتيب النتائج تصاعديًا حسب عمود student_name.

- في حال وجود عمود بالاسم نفسه في جداول الاستعلام يجب كتابة اسم الجدول قبل اسم العمود (students.class_id)، لتجنب ظهور رسالة خطأ ambiguous column name.
- يُمكن ترتيب بيانات الطلاب تصاعديًا | تنازليًا باستخدام العبارة
  ORDER BY student_name ASC | DESC
  مع العلم بأن الترتيب يكون تصاعديًا بشكل افتراضي عند عدم تحديد نوع الترتيب.

```
rows = cursor.fetchall()
```

- استرجاع جميع نتائج الاستعلام.

```
for row in rows:
    print(row)
```

- طباعة كل صف من النتائج في سطر مستقل باستخدام الحلقة التكرارية.

```
connection.close()
```

- إغلاق الاتصال بقاعدة البيانات.

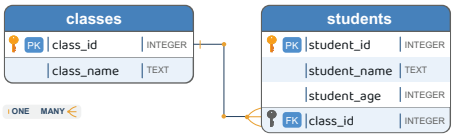
Program Output

```
Enter class name: 11S1
('Fahad Mohammed Ali', 14)
('Talal Saad Saleh', 15)
```

الاستعلام باستخدام GROUP BY

مثال 2

الاستعلام عن بيانات مُجمعة بقاعدة البيانات school_data.db.



شكل (4-4) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

المطلوب: كتابة التعليمة البرمجية لتنفيذ التالي:

- الاستعلام عن إجمالي عدد الطلاب لكل فصل.

خطوات التنفيذ

- استيراد حزمة sqlite3.
- إنشاء/ فتح اتصال بقاعدة البيانات.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- استرجاع البيانات.
- طباعة النتائج.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة النتائج.

```
import sqlite3

connection = sqlite3.connect('school_data.db')
cursor = connection.cursor()

cursor.execute('''
    SELECT class_name, COUNT (student_name)
    FROM students
    INNER JOIN classes
        ON students.class_id = classes.class_id
    WHERE class_name = ?
    GROUP BY class_name '''
)

rows=cursor.fetchall()
for row in rows:
    print(row)

connection.close()
```

التفسيـــــر

import sqlite3

- استيراد الحزمة sqlite3.

---

connection = sqlite3.connect('school_data.db')

- الاتصال بقاعدة بيانات school_data.db.

---

cursor = connection.cursor()

- إنشاء كائن المؤشر cursor.

```
cursor.execute('''
        SELECT class_name, COUNT (student_name)
        FROM students
        INNER JOIN classes
                ON  students.class_id = classes.class_id
        WHERE class_name = ?
        GROUP BY class_name '''
)
```

استخدام:

- الدالة COUNT لحساب عدد الطلاب في كل فصل.

يُوجد العديد من الدوال التي يُمكن استخدامها للاستعلام مثل:

(المتوسط الحسابي avg، أكبر قيمة max، أصغر قيمة min، الجمع sum).

- عبارة INNER JOIN لدمج الجدولين students و classes بناءً على العمود المشترك.
- عبارة GROUP BY لتجميع النتائج حسب كل فصل.

```
rows = cursor.fetchall()
```

- استرجاع جميع نتائج الاستعلام.

```
for row in rows:
    print(row)
```

- طباعة كل صف من النتائج في سطر مستقل باستخدام الحلقة التكرارية.

```
connection.close()
```

- إغلاق الاتصال بقاعدة البيانات.

Program Output

```
('11A1', 3)
('11S1', 2)
('11S2', 2)
('11S3', 1)
```

يُمكن ترتيب البيانات تنازلياً حسب إجمالي عدد طلاب كل فصل باستخدام عبارة ORDER BY.

مثال:

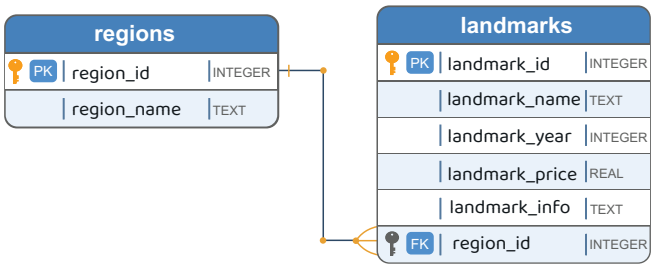
```
SELECT class_name, COUNT(student_name) AS total_student
FROM students
INNER JOIN classes
        ON  students.class_id = classes.class_id
GROUP BY class_name
ORDER BY total_student DESC
```

- تُستخدم الدالة COUNT لحساب إجمالي عدد الطلاب.
- تُستخدم الكلمة المفتاحية AS لإعطاء اسم مستعار مؤقت لناتج الدالة COUNT.

أوراق العمل

ورقة عمل (5) التاريخ:

الاستعلام عن بيانات محددة من الجدول landmarks مرتبة ترتيباً تصاعدياً حسب عمود landmark_year بقاعدة البيانات Kuwait_landmarks.db.



شكل (4-5) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

1. استدع المشروع select1_landmark.
2. افتح ملف data_select.py.

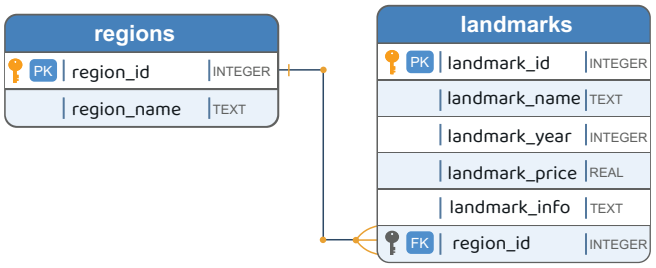
```
import sqlite3
# Get region name from user
var_region_name = input('Enter region name: ')          # Kuwait City

connection = sqlite3.connect('Kuwait_landmarks.db')
cursor = connection.cursor()

# Retrieve data from a specific region
cursor.execute('''
        SELECT landmark_name, landmark_year, region_name
        FROM landmarks
        INNER JOIN regions
                ON ..... = .....
        WHERE region_name = ?
        ORDER BY ..... ''',
        (var_region_name,)
)
```

```
18
19      # Print each row of the results on a separate line using a loop
20      rows = cursor.fetchall()
21      for row in rows:
22              print(row)
23
24      connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:
   - ◄ استعلم عن جميع سجلات الجدول landmarks حسب القيمة المُدخلة لعمود region_name، مرتبة ترتيباً تصاعدياً حسب عمود landmark_year.
4. نفذ التعليمات البرمجية.
5. عاين نتائج الاستعلام.

ورقة عمل (6) التاريخ:

الاستعلام عن بيانات عدد المعالم مُجمعة حسب المنطقة في الجدول landmarks بقاعدة البيانات Kuwait_landmarks.db.



شكل (4-6) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

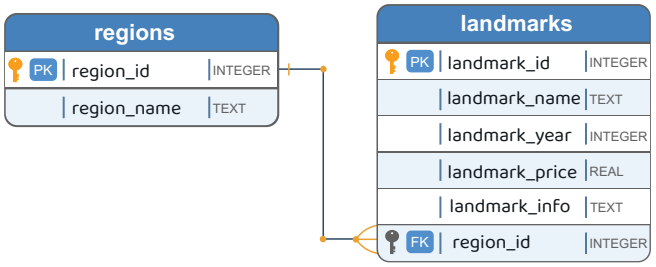
1. استدع المشروع landmark_select2.
2. افتح ملف data_select.py.

```
1   import sqlite3
2
3   connection = sqlite3.connect('Kuwait_landmarks.db')
4   cursor = connection.cursor()
5
6   # Retrieve landmarks count in each region
7   cursor.execute('''
8           SELECT region_name, ……
9           FROM landmarks
10          INNER JOIN regions
11                  ON landmarks.region_id = regions.region_id
12          …… …… region_name
13  ''')
14
15  # Print each row of the results on a separate line using a loop
16  rows = cursor.fetchall()
17  for row in rows:
18          print(row)
19
20  connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:
   ◄ استعلم عن اسم المنطقة، وعدد المعالم مُجمعة حسب المنطقة.
4. نفذ التعليمات البرمجية.
5. عاين نتائج الاستعلام.

ورقة عمل (7) التاريخ:

الاستعلام عن متوسط أسعار دخول المعالم المدفوعة لكل منطقة مع ترتيب النتائج ترتيبًا تنازليًا حسب متوسط الأسعار بقاعدة البيانات Kuwait_landmarks.db.



شكل (4-7) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

1. استدع المشروع landmark_select3.
2. افتح ملف data_select.py.

```
import sqlite3

connection = sqlite3.connect('Kuwait_landmarks.db')
cursor = connection.cursor()

# Retrieve the average entry fees.
cursor.execute('''
        SELECT region_name, AVG(.....) AS average_price
        FROM landmarks
        INNER JOIN regions
                ON landmarks.region_id = regions.region_id
        WHERE .....
        GROUP BY .....
        ORDER BY ..... DESC

''')

# Print each row of the results on a separate line using a loop
rows = cursor.fetchall()
for row in rows:
        print(row)

connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:
   ◄ استعلم عن اسم المنطقة، وحساب متوسط أسعار دخول المعالم المدفوعة، مع تجميع النتائج حسب المنطقة وترتيبها ترتيبًا تنازليًا وفقًا لمتوسط الأسعار.
4. نفذ التعليمات البرمجية.
5. عاين نتائج الاستعلام.

تحديث البيانات في الجداول المترابطة
Updating Data in Related Tables

نتاجات التعلم

- تعريف وظيفة تعليمة UPDATE ودورها في تعديل البيانات داخل صفوف موجودة في جدول محدد.
- تفسير أثر تعديل المفتاح الأساسي في جدول الأب Parent على الصفوف المرتبطة في جدول الابن Child في العلاقات العلائقية.
- تمييز إجراءات التكامل المرجعي ON UPDATE، ومعرفة استخدام كل إجراء.
- تطبيق شروط التعديل بشكل صحيح لضمان عدم تعديل صفوف غير مستهدفة داخل الجدول.
- تنفيذ تعليمة UPDATE في SQLite لتعديل قيم أعمدة معينة وفق شرط محدد.
- برمجة تحديث البيانات باستخدام Python عبر حزمة sqlite3، مع تفعيل دعم المفاتيح الخارجية لمنع الأخطاء.
- تحليل عدد الصفوف التي تم تحديثها من خلال استخدام connection.total_changes أثناء تنفيذ عمليات التعديل.
- معاينة التغييرات على الجداول بعد التحديث باستخدام DB Browser for SQLite للتحقق من نجاح العملية.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم

تحديث البيانات في قواعد البيانات العلائقية Updating Data

تُستخدم عملية التحديث Update لتعديل قيمة أو أكثر داخل صفوف موجودة في جدول محدد، وفق شروط تُحدد الصفوف المطلوب تحديثها لضمان تغيير البيانات المطلوبة فقط.
عند تعديل قيمة المفتاح الأساسي في جدول الأب Parent في أي علاقة يؤثر بالتبعية على الصفوف المرتبطة في جدول الابن Child، بينما تعديل أي قيمة في جدول الابن Child لا يؤثر على جدول الأب Parent.

لاحظ

تغيير قيمة المفتاح الأساسي غير مُستحسن لأنه يجب أن يبقى ثابتًا، حفاظًا على سلامة البيانات وتكامل العلاقات بين الجداول.

إجراءات التعديل عند إنشاء القيد FOREIGN KEY

هُناك مجموعة من الإجراءات يُمكن تفعيلها عند إنشاء القيد FOREIGN KEY، والشكل التالي يوضح جدولين مرتبطين قبل إجراء أي تعديل.

جدول الطلاب students (جدول الابن Child)

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
3	Meshari Ahmed Saud	14	102
4	Saud Abdullah Yousef	15	102
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes (جدول الأب Parent)

class_id	class_name
101	11S1
102	11S2
103	11A1

الشكل (5-1) يمثل جدولين مرتبطين قبل إجراء أي تعديل.

ON UPDATE CASCADE:

عند تعديل قيمة المفتاح الأساسي في جدول الأب Parent، تُحدث قاعدة البيانات قيم المفتاح الخارجي المرتبطة في جدول الابن Child تلقائياً.

جدول الطلاب students

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
3	Meshari Ahmed Saud	14	**104**
4	Saud Abdullah Yousef	15	**104**
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes

class_id	class_name
101	11S1
104	11S2
103	11A1

الشكل (5-2) يُمثل الجدولين بعد تحديث قيمة رقم الفصل class_id لتصبح 104 بدلاً من 102، وتحديث جميع الصفوف المرتبطة به من جدول students تلقائياً.

ON UPDATE SET NULL:

عند تعديل قيمة المفتاح الأساسي في جدول الأب Parent، تُحدّث قاعدة البيانات قيم المفتاح الخارجي المرتبطة في جدول الابن Child إلى NULL تلقائياً.

جدول الطلاب students

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
3	Meshari Ahmed Saud	14	**NULL**
4	Saud Abdullah Yousef	15	**NULL**
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes

class_id	class_name
101	11S1
104	11S2
103	11A1

الشكل (5-3) يُمثل الجدولين بعد تحديث قيمة رقم الفصل class_id لتصبح 104 بدلاً من 102، وتخصيص القيمة NULL للمفتاح الخارجي للصفوف المرتبطة تلقائيًا.

يعمل الإجراء بشكل صحيح ما لم يكن مطبّقًا على عمود المفتاح الخارجي قيد NOT NULL، قد يؤدي ذلك إلى ظهور صفوف في جدول الابن غير مرتبطة بأي صف في جدول الأب، ويُطلق عليها Orphan Rows.

ON UPDATE NO ACTION:

عند تعديل قيمة المفتاح الأساسي في جدول الأب Parent، تمنع قاعدة البيانات تعديل القيمة لارتباطها بصفوف أخرى في جدول الابن Child.

جدول الطلاب students

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
3	Meshari Ahmed Saud	14	102
4	Saud Abdullah Yousef	15	102
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes

class_id	class_name
101	11S1
102	11S2
103	11A1

الشكل (5-4) يُمثل الجدولين بعد فشل محاولة تحديث قيمة رقم الفصل class_id لتصبح 104 بدلًا من 102 لتواجد صفوف مرتبطة.

في قواعد البيانات العلائقية SQLite، يُعد إجراء ON UPDATE CASCADE بشكل عام الإجراء الأكثر شيوعًا والموصى به للحفاظ على اتساق البيانات وسلامتها.

صيغة التعليمة البرمجية لتحديث البيانات SQLite في Python

```
cursor.execute('''
UPDATE table_name
SET column_name = ?
WHERE condition_column = ?''',
(value, condition_value)
)
```

يُنفذ أمر SQL داخل قاعدة البيانات

اسم الجدول

أمر SQL لتحديث البيانات

عُنصر مُؤقت placeholder للقيمة الجديدة

اسم العمود المطلوب تحديثه

عُنصر مُؤقت placeholder لقيمة الشرط

العمود المُستخدم في جملة الشرط

القيمة الجديدة

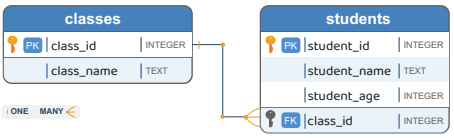
قيمة الشرط

شكل (5-5) يُمثل صيغة كتابة التعليمة البرمجية لتحديث البيانات.

تعديل صفوف في جدول الأب Parent

مثال

تحديث بيانات عمود في جدول classes قاعدة بيانات school_data.db



شكل (5-6) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

المطلوب: كتابة التعليمة البرمجية لتنفيذ التالي:

- تعديل عمود class_id لتصبح قيمته 122.
- تعديل عمود class_name لتصبح قيمته 11Science2.
- وفقًا للشرط (class_id = 102).

تم استخدام ON UPDATE CASCADE عند إنشاء العلاقة بين الجدولين.

خطوات التنفيذ

- معاينة بيانات الجدول students في برنامج DB Browser for SQLite.
- استيراد حزمة sqlite3.
- استقبال رقم الفصل الحالي.
- استقبال رقم الفصل الجديد.
- استقبال اسم الفصل الجديد.
- إنشاء/ فتح اتصال بقاعدة البيانات.
- تفعيل دعم المفاتيح الخارجية.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- تعديل البيانات.
- طباعة عدد الصفوف التي تم تحديثها.
- حفظ التغييرات.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة بيانات الجدول students في برنامج DB Browser for SQLite.

```
import sqlite3

var_old_class_id = int(input('Enter old class id: '))          # 102
var_new_class_id = int(input('Enter new class id: '))          # 122
var_new_class_name = input('Enter new class name: ')   #11Science2

connection = sqlite3.connect('school_data.db')
connection.execute('PRAGMA foreign_keys = ON')

cursor = connection.cursor()

cursor.execute('''
    UPDATE classes
    SET class_id=?,class_name=?
    WHERE class_id=? ''',
    (var_new_class_id, var_new_class_name, var_old_class_id)
)

print('Number of rows updated = ', connection.total_changes)

connection.commit()
connection.close()
```

التفسيـــــر

```
import sqlite3
```

- استيراد الحزمة sqlite3.

```
var_old_class_id = int(input('Enter old class id: '))          # 102
var_new_class_id = int(input('Enter new class id: '))          # 122
var_new_class_name = input('Enter new class name: ')   # 11Science2
```

- استقبال من المُستخدم القيم المطلوبة.

```
connection = sqlite3.connect('school_data.db')
```

- الاتصال بقاعدة بيانات school_data.db.

```
connection.execute('PRAGMA foreign_keys = ON')
```

- تفعيل دعم المفاتيح الخارجية.

```
cursor = connection.cursor()
```

- إنشاء كائن المؤشر cursor.

```
cursor.execute('''
        UPDATE classes
         SET class_id = ?, class_name = ?
         WHERE class_id = ? ''',
         (var_new_class_id, var_new_class_name, var_old_class_id)
)
```

- تحديث عمودي class_id و class_name لجدول classes اعتمادًا على القيم var_new_class_id و var_new_class_name و var_old_class_id التي أدخلها المُستخدم، سيتم تحديث القيم المرتبطة تلقائيًا في جدول student بسبب وجود الإجراء ON UP-DATE CASCADE عند إنشاء الجدول.

```
print('Number of rows updated = ', connection.total_changes)
```

- طباعة إجمالي عدد الصفوف التي تم تحديثها في الجداول المُترابطة بقاعدة البيانات نتيجة عملية UPDATE.

```
connection.commit()
```

- حفظ التغييرات.

```
connection.close()
```

- إغلاق الاتصال بقاعدة البيانات.

Program Output

```
Enter old class id: 102
Enter new class id: 122
Enter new class name: 11Science2
Number of rows updated: 3
```

تم تحديث صف واحد مباشرةً في جدول classes (جدول الأب)، ونتج عن ذلك تحديث صفّين تلقائيًا في جدول students (جدول الابن) بسبب إجراء ON UPDATE CASCADE عند إنشاء الجدول ليصبح إجمالي الصفوف المتأثرة 3 صفوف.

معاينة بيانات الجداول في برنامج DB Browser for SQLite

جدول classes قبل التعديل

Table: classes

	class_id	class_name
	Filter	Filter
1	101	11S1
2	102	11S2
3	103	11A1
4	104	11A2
5	105	11S3

جدول classes بعد التعديل

Table: classes

	class_id	class_name
	Filter	Filter
1	101	11S1
2	103	11A1
3	104	11A2
4	105	11S3
5	122	11Science2

شكل (5-7) يُوضح بيانات جدول classes قبل وبعد التعديل.

جدول students قبل التعديل

Table: students

	student_id	student_name	student_age	class_id
	Filter	Filter	Filter	Filter
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	102
4	4	Saud Abdullah Yousef	15	102
5	5	Fahad Farhan Ahmed	15	103
6	6	Abdullateef Ali Hussain	16	103
7	7	Ahmed Ali Saad	16	103
8	8	Bader Abdullah Yousef	15	105

جدول students بعد التعديل

Table: students

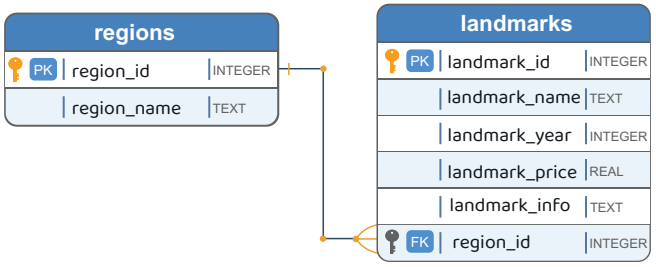
	student_id	student_name	student_age	class_id
	Filter	Filter	Filter	Filter
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	122
4	4	Saud Abdullah Yousef	15	122
5	5	Fahad Farhan Ahmed	15	103
6	6	Abdullateef Ali Hussain	16	103
7	7	Ahmed Ali Saad	16	103
8	8	Bader Abdullah Yousef	15	105

شكل (5-8) يُوضح بيانات جدول students قبل وبعد التعديل.

أوراق العمل

ورقة عمل (8) التاريخ:

تعديل بيانات جدول regions بقاعدة البيانات Kuwait_landmarks.db.



شكل (5-9) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

1. استدع المشروع landmark_update.
2. افتح ملف data_update.py.

```
1   import sqlite3
2   # Get old and new region id from user
3   var_old_region_id = int(input('Enter old region id: '))      # 206
4   var_new_region_id = int(input('Enter new region id: '))      # 106
5
6   connection = sqlite3.connect('Kuwait_landmarks.db')
7   connection.execute('PRAGMA foreign_keys = ON')
8   cursor = connection.cursor()
9
10  # update region's id
11  cursor.execute('''
12          ..... regions
13          ..... region_id=?
14          ..... region_id=?
15          ''',
16          (var_new_region_id, var_old_region_id)
17  )
18  print('Number of rows updated = ', connection.total_changes)
```

```
19
20      connection.commit()
21      connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:
   ◄ حـدث بيانـات الصـف (region_id = 206) لتصبـح قيمـة العمـود region_id تسـاوي 106 فـي الجـدول regions.
4. نفذ التعليمات البرمجية.
5. عاين بيانات الجداول في برنامج DB Browser for SQLite.

جدول regions قبل التعديل

Database Structure | Browse Data

Table: regions

	region_id	region_name
1	101	Kuwait City
2	102	Al-Rai
3	103	Failaka Island
4	104	South Kuwait
5	105	Sharq
6	206	Al-Murqab

جدول regions بعد التعديل

Database Structure | Browse Data

Table: regions

	region_id	region_name
1	101	Kuwait City
2	102	Al-Rai
3	103	Failaka Island
4	104	South Kuwait
5	105	Sharq
6	106	Al-Murqab

شكل (5-10) يُوضح بيانات جدول regions قبل وبعد التعديل.

جدول landmarks قبل التعديل

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: landmarks

	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum houses...	206
2	2	Failaka Island	1950	5.0	An island with ...	103
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional ...	206
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable ...	104
5	5	Kuwait National Library	1994	0.0	A prominent ...	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the largest...	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque ...	101
8	8	Al-Tahrir Tower	1996	0.0	NULL	101
9	9	Museum of Modern Art	1980	1.0	NULL	105

جدول landmarks بعد التعديل

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: landmarks

	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum house...	106
2	2	Failaka Island	1950	5.0	An island with ...	103
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional ...	106
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable ...	104
5	5	Kuwait National Library	1994	0.0	A prominent ...	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the larges...	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque...	101
8	8	Al-Tahrir Tower	1996	0.0	NULL	101
9	9	Museum of Modern Art	1980	1.0	NULL	105

شكل (5-11) يُوضح بيانات جدول landmarks قبل وبعد التعديل.

حذف البيانات من الجداول المترابطة
Deleting Data from Related Tables

نتاجات التعلم

- تعريف وظيفة تعليمة DELETE ودورها في حذف صف واحد أو أكثر من جدول محدد داخل قاعدة البيانات.
- تمييز أثر حذف صف من جدول الأب Parent على الصفوف المرتبطة في جدول الابن Child وفق قواعد التكامل المرجعي.
- تفسير إجراءات ON DELETE، ومعرفة متى يستخدم كل إجراء منها.
- تطبيق شروط الحذف بشكل دقيق لضمان عدم حذف صفوف غير مقصودة.
- تنفيذ تعليمة DELETE في SQLite لحذف بيانات وفق شرط محدد (WHERE).
- برمجة عملية الحذف باستخدام Python عبر حزمة sqlite3 مع تفعيل دعم المفاتيح الخارجية لتطبيق القواعد بشكل صحيح.
- تحليل عدد الصفوف المحذوفة باستخدام connection.total_changes أثناء تنفيذ عملية الحذف.
- معاينة بيانات الجداول قبل وبعد عملية الحذف باستخدام DB Browser for SQLite للتحقق من نجاح العملية.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم

حذف البيانات في قواعد البيانات العلائقية Deleting Data

تُستخدم عملية الحذف DELETE لحذف صفوف محددة من جدول وفق شروط مُحددة.

عند حذف صف من جدول الأب Parent في أي علاقة يؤثر بالتبعية على الصفوف المرتبطة في جدول الابن Child، بينما حذف أي صف في جدول الابن Child لا يؤثر على جدول الأب Parent.

إجراءات الحذف عند إنشاء القيد FOREIGN KEY

هناك مجموعة من الإجراءات يمكن تفعيلها عند إنشاء القيد FOREIGN KEY منها:

جدول الطلاب students (جدول الابن Child)

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
3	Meshari Ahmed Saud	14	102
4	Saud Abdullah Yousef	15	102
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes (جدول الأب Parent)

class_id	class_name
101	11S1
102	11S2
103	11A1

الشكل (6-1) يُمثل جدولين مرتبطين.

ON DELETE CASCADE:

عند حذف صف من جدول الأب Parent، تحذف قاعدة البيانات جميع الصفوف المرتبطة بالمفتاح الخارجي في جدول الابن Child تلقائياً.

جدول الطلاب students

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes

class_id	class_name
101	11S1
103	11A1

الشكل (6-2) يُمثل الجدولين بعد حذف الصف الذي فيه قيمة رقم الفصل class_id تساوي 102 من جدول classes، وحذف جميع الصفوف المرتبطة به من جدول students تلقائياً .

ON DELETE SET NULL:

عند حذف صف من جدول الأب Parent، تُحدث قاعدة البيانات قيم المفتاح الخارجي المرتبطة في جدول الابن Child إلى NULL تلقائياً.

جدول الطلاب students

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
3	Meshari Ahmed Saud	14	NULL
4	Saud Abdullah Yousef	15	NULL
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes

class_id	class_name
101	11S1
103	11A1

الشكل (6-3) يُمثل الجدولين بعد حذف الصف الذي فيه قيمة رقم الفصل class_id تساوي 102 من جدول classes، وتخصيص القيمة NULL للمفتاح الخارجي للصفوف المرتبطة تلقائيًا.

ON DELETE NO ACTION:

عند حذف صف من جدول الأب Parent، تمنع قاعدة البيانات إجراء الحذف لارتباطها بصفوف أخرى في جدول الابن Child.

جدول الطلاب students

student_id	student_name	student_age	class_id
1	Fahad Mohammed Ali	14	101
2	Talal Saad Saleh	15	101
3	Meshari Ahmed Saud	14	102
4	Saud Abdullah Yousef	15	102
5	Fahad Farhan Ahmed	15	103
6	Abdullateef Ali Hussain	16	103

جدول الفصول classes

class_id	class_name
101	11S1
102	11S2
103	11A1

الشكل (6-4) يُمثل الجدولين بعد فشل محاولة حذف الصف الذي فيه قيمة رقم الفصل class_id تساوي 102 من جدول classes لتواجد صفوف مرتبطة.

في قواعد البيانات العلائقية SQLite، يُعد إجراء ON DELETE CASCADE بشكل عام الإجراء الأكثر شيوعًا والمُوصى به للحفاظ على اتساق البيانات وسلامتها.

صيغة التعليمة البرمجية لحذف البيانات SQLite في Python

يُنفذ أمر SQL
داخل قاعدة البيانات

اسم الجدول

عُنصر مؤقت placeholder
لقيمة الشرط

```
cursor.execute ("DELETE FROM table_name WHERE column_name = ?", (value))
```

أمر SQL لحذف
صف/ صفوف من الجدول

العمود المُستخدم في
جملة الشرط

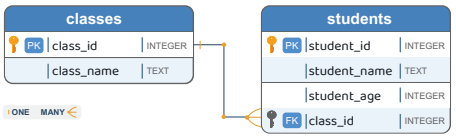
قيمةالشرط

شكل (6-5) يُمثل صيغة كتابة التعليمة البرمجية لحذف الصفوف.

حذف صفوف في جدول الأب Parent

مثال

حذف الصفوف من جدول بقاعدة البيانات school_data.db.



شكل (6-6) يُمثل مخطط ERD لقاعدة بيانات الطلاب.

المطلوب:

- حذف الصفوف من جدول classes.
- وفقاً للشرط ('class_name = '11A1).

تم استخدام ON DELETE CASCADE عند إنشاء العلاقة بين الجدولين.

خطوات التنفيذ

- معاينة بيانات الجدول students في برنامج DB Browser for SQLite.
- استيراد حزمة sqlite3.
- استقبال الفصل وتخزينه في متغير.
- إنشاء/ فتح اتصال بقاعدة البيانات.
- تفعيل دعم المفاتيح الخارجية.
- إنشاء كائن المؤشر للتعامل مع قاعدة البيانات.
- حذف البيانات.

- طباعة عدد الصفوف التي تم حذفها.
- حفظ التغييرات.
- إغلاق الاتصال.
- تنفيذ التعليمات البرمجية.
- معاينة بيانات الجدول students في برنامج DB Browser for SQLite.

```
1   import sqlite3
2
3   var_class_name = input('Enter class name: ')      # 11A1
4
5   connection = sqlite3.connect('school_data.db')
6   connection.execute('PRAGMA foreign_keys = ON')
7   cursor = connection.cursor()
8
9   cursor.execute('''
10      DELETE FROM classes WHERE class_name = ?''',
11      (var_class_name,)
12  )
13  print('Number of rows deleted = ', connection.total_changes)
14
15  connection.commit()
16  connection.close()
```

التفسيـــــر

import sqlite3

- استيراد الحزمة sqlite3.

---

var_class_name = input('Enter class name: ') #11A1

- استقبال بيانات من المُستخدم.

---

connection = sqlite3.connect('school_data.db')

- الاتصال بقاعدة بيانات school_data.db.

---

connection.execute('PRAGMA foreign_keys = ON')

- تفعيل دعم المفاتيح الخارجية.

```
cursor = connection.cursor()
```

إنشاء كائن المؤشر cursor.

```
cursor.execute('''
        DELETE FROM classes WHERE class_name = ?
        ''',( var_class_name,)
```

حذف الصف من جدول classes عندما تكون قيمة العمود class_name تساوي قيمة المتغير var_class_name، سيتم حذف الصفوف المرتبطة تلقائيًا في جدول student بسبب وجود الإجراء ON DELETE CASCADE عند إنشاء الجدول.

```
print('Number of rows deleted = ', connection.total_changes)
```

طباعة إجمالي عدد الصفوف التي تم حذفها في الجداول المُترابطة بقاعدة البيانات نتيجة عملية DELETE.

```
connection.commit()
```

حفظ التغييرات.

```
connection.close()
```

إغلاق الاتصال بقاعدة البيانات.

Program Output

```
Enter class name: 11A1
Number of rows deleted: 4
```

تم حذف صف واحد مباشرةً في جدول classes (جدول الأب)، ونتج عن ذلك تحديث ثلاثة صفوف تلقائيًا في جدول students (جدول الابن) بسبب إجراء ON DELETE CASCADE عند إنشاء الجدول ليصبح إجمالي الصفوف المتأثرة 4 صفوف.

معاينة بيانات الجداول في برنامج DB Browser for SQLite

جدول classes قبل الحذف

Table: classes

	class_id	class_name
	Filter	Filter
1	101	11S1
2	103	11A1
3	104	11A2
4	105	11S3
5	122	11Science2

جدول classes بعد الحذف

Table: classes

	class_id	class_name
	Filter	Filter
1	101	11S1
2	104	11A2
3	105	11S3
4	122	11Science2

شكل (6-7) يُوضح بيانات جدول classes قبل وبعد الحذف.

جدول students قبل الحذف

Table: students

	student_id	student_name	student_age	class_id
	Filter	Filter	Filter	Filter
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	122
4	4	Saud Abdullah Yousef	15	122
5	5	Fahad Farhan Ahmed	15	103
6	6	Abdullateef Ali Hussain	16	103
7	7	Ahmed Ali Saad	16	103
8	8	Bader Abdullah Yousef	15	105

جدول students بعد الحذف

Table: students

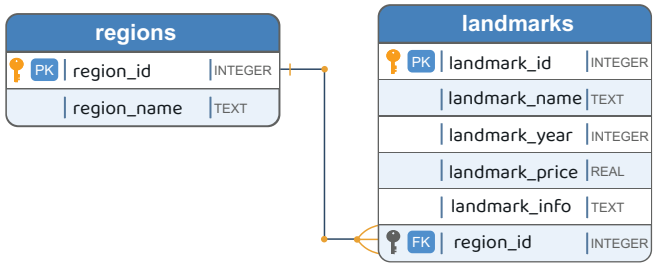
	student_id	student_name	student_age	class_id
	Filter	Filter	Filter	Filter
1	1	Fahad Mohammed Ali	14	101
2	2	Talal Saad Saleh	15	101
3	3	Meshari Ahmed Saud	14	122
4	4	Saud Abdullah Yousef	15	122
5	8	Bader Abdullah Yousef	15	105

شكل (6-8) يُوضح بيانات جدول students قبل وبعد الحذف.

ورقة عمل (9)

التاريخ:

حذف صف من جدول regions بقاعدة البيانات Kuwait_landmarks.db.



شكل (6-9) يُمثل مخطط ERD لقاعدة بيانات معالم الكويت.

تطبيق الخطوات التالية بالاسترشاد بالمخطط أعلاه:

1. استدع المشروع landmark_delete.
2. افتح ملف data_delete.py.

```
import sqlite3
# Get region id from user
var_region_id = int(input('Enter region id: '))                # 103

connection = sqlite3.connect('Kuwait_landmarks.db')
connection.execute('PRAGMA foreign_keys = ON')
cursor = connection.cursor()

# Delete region's id
cursor.execute('''
        ..... FROM regions
        ..... region_id=?
        ''',
        (var_region_id,)
)
print('Number of rows deleted = ', connection.total_changes)
```

```
17
18      connection.commit()
19      connection.close()
```

3. استكمل التعليمات البرمجية لتنفيذ التالي:
   ◄ احذف الصف (region_id=103) من جدول regions.
4. نفذ التعليمات البرمجية.
5. عاين بيانات الجداول في برنامج DB Browser for SQLite.

جدول regions قبل الحذف

Database Structure | Browse Data

Table: regions

	region_id	region_name
1	101	Kuwait City
2	102	Al-Rai
3	103	Failaka Island
4	104	South Kuwait
5	105	Sharq
6	106	Al-Murqab

جدول regions بعد الحذف

Database Structure | Browse Dat

Table: regions

	region_id	region_name
1	101	Kuwait City
2	102	Al-Rai
3	104	South Kuwait
4	105	Sharq
5	106	Al-Murqab

شكل (6-10) يُوضح بيانات جدول regions قبل وبعد الحذف.

جدول landmarks قبل الحذف

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: landmarks

	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum house…	206
2	2	Failaka Island	1950	5.0	An island with …	103
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional …	206
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable …	104
5	5	Kuwait National Library	1994	0.0	A prominent …	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the larges…	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque…	101
8	8	Al-Tahrir Tower	1996	0.0	NULL	101
9	9	Museum of Modern Art	1980	1.0	NULL	105

جدول landmarks بعد الحذف

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: landmarks

	landmark_id	landmark_name	landmark_year	landmark_price	landmark_info	region_id
1	1	Kuwait National Museum	1983	2.5	The museum house…	206
3	3	Souq Al-Mubarakiya	1897	0.0	A traditional …	206
4	4	Sheikh Jaber Al-Ahmad Bridge	2019	0.0	A remarkable …	104
5	5	Kuwait National Library	1994	0.0	A prominent …	101
6	6	Sheikh Jaber Al-Ahmad Cultural Center	2016	0.0	It is the larges…	101
7	7	Grand Mosque of Kuwait	1986	0.0	The Grand Mosque…	101
8	8	Al-Tahrir Tower	1996	0.0	NULL	101
9	9	Museum of Modern Art	1980	1.0	NULL	105

شكل (6-11) يُوضح بيانات جدول landmarks قبل وبعد الحذف.

الوحـدة الثانية- المُنتجات الرقمية Digital Products

1. الذكاء الاصطناعي وتكامله مع قواعد البيانات
   مشروع تقنيات كشف الوجوه
2. مراحل تصميم وتطوير المنتج الرقمي

المنتجات الرقمية
الذكاء الاصطناعي وتكامله مع قواعد البيانات
AI and Database Integration

نتاجات التعـلم

- شرح المفاهيم الأساسية لتقنيات الذكاء الاصطناعي ورؤية الحاسوب.
- الربط بين خوارزميات التعرف على الوجوه وقواعد البيانات في مشروع تطبيقي.
- إنشاء برنامج يستخدم حزم Python لتسجيل وتحليل البيانات البيومترية.

يمثـل رمـز الاسـتجابة السـريعة QR رابـط
لملفـات أوراق العمـل، ومصـادر التعلـم

مشروع توظيف تقنيات الذكاء الاصطناعي مع قواعد البيانات العلائقية

يهدف هذا المشروع إلى توظيف تقنيات الذكاء الاصطناعي مع قواعد البيانات العلائقية لبناء نظام ذكي قادر على تخزين البيانات وتنظيمها وتحليلها بكفاءة. ويعتمد المشروع على ربط الخوارزميات الذكية ببيانات مترابطة، وتحسين دقة النتائج.

- **مفهوم الذكاء الاصطناعي وتطبيقاته:**

الذكاء الاصطناعي هو فرع من علوم الحاسوب يهدف إلى تمكين الأجهزة والبرامج من محاكاة بعض قدرات الإنسان مثل التعلم، والتفكير، واتخاذ القرار. ويُستخدم في تطبيقات متعددة مثل التعرف على الوجوه، وتحليل الصور، والمساعدات الذكية، والأنظمة التنبؤية.

- **مفهوم قواعد البيانات العلائقية:**

قواعد البيانات العلائقية هي نظم تُستخدم لتخزين البيانات وتنظيمها داخل جداول مترابطة فيما بينها بعلاقات محددة. تعتمد هذه القواعد على المفاتيح الأساسية والخارجية لضمان سلامة البيانات وسهولة استرجاعها.

- **أهمية ربط الذكاء الاصطناعي بقواعد البيانات:**

تكمن أهمية ربط الذكاء الاصطناعي بقواعد البيانات في تمكين الأنظمة الذكية من حفظ البيانات الناتجة عن التحليل والمعالجة بشكل منظم. ويساعد هذا الربط على تحسين دقة النتائج، وإمكانية الرجوع إلى البيانات، وتطوير أداء الخوارزميات بمرور الوقت.

- **تصميم قاعدة بيانات للتطبيقات الذكية:**

يتطلب تصميم قاعدة بيانات للتطبيقات الذكية تحديد الجداول المناسبة وأنواع البيانات والعلاقات بينها بما يتوافق مع طبيعة التطبيق.

- **آلية تخزين بيانات الذكاء الاصطناعي في الجداول المترابطة:**

يتم تخزين بيانات الذكاء الاصطناعي، مثل نتائج التنبؤ أو بيانات التعرف Recognition Data، داخل جداول مرتبطة ببعضها من خلال علاقات منطقية. ويسمح ذلك بتنظيم البيانات وربطها بمعلومات أخرى.

- **توظيف الاستعلامات والتقارير في دعم اتخاذ القرار:**

تُستخدم الاستعلامات لاستخراج البيانات وتحليلها من قواعد البيانات العلائقية بصورة منظمة ودقيقة. كما تُسهم التقارير في تلخيص النتائج وعرض المعلومات المهمة بشكل واضح، مما يدعم اتخاذ القرار اعتمادًا على بيانات صحيحة ومترابطة.

مشروع: ربط خوارزمية ذكية بقاعدة بيانات علائقية للتعرف على الوجوه

يهدف هذا المشروع إلى ربط خوارزمية ذكية للتعرّف على الوجوه بقاعدة بيانات علائقية، حيث يتم استخدام صور ثابتة للتعرّف على الوجوه في المرحلة الأولى. ثم يتم تطوير البرنامج ليعمل عبر الكاميرا في الزمن الحقيقي، مع تخزين بيانات الوجوه ونتائج التعرّف داخل قاعدة البيانات، وطباعة تقارير توضح عمليات التعرّف والمعلومات المرتبطة بكل وجه لدعم التحليل واتخاذ القرار.

الحزم المستخدمة في المشروع

الحزمة	الاستخدام	تثبيت الحزمة
OpenCV	اكتشاف الوجوه والتعامل مع الصور.	pip install opencv-python
numpy	التعامل مع المصفوفات والبيانات العددية وتنفيذ العمليات الرياضية بكفاءة عالية.	pip install numpy كذلك يتم تثبيت حزمة numpy بشكل تلقائي عند تثبيت حزمة opencv.
reportlab	إنشاء التقارير والملفات بصيغة PDF برمجيًا بطريقة مرنة ومنسقة.	pip install reportlab
os	إدارة الملفات ومسارات الصور.	حزمة مدمجة في Python
sqlite3	لإنشاء وتخزين البيانات في قاعدة بيانات محلية.	
datetime	تسجيل الوقت والتاريخ لكل وجه يتم التعرف عليه.	

جدول رقم (7-1) يُوضح أسماء الحزم المُستخدمة في المشروع

مراحل المشروع

استيراد الحزم وتجهيز البيئة

يتم استيراد الحزم المستخدمة في المشروع.

إعداد قاعدة البيانات العلائقية

يتم إنشاء قاعدة بيانات SQLite تحتوي على جداول لتخزين بيانات الوجوه وسجلات التعرف (الاسم، وقت الاكتشاف).

إدخال الصورة ومعالجتها مبدئيًا

تُحمّل الصورة المدخلة من المستخدم وتُحوّل إلى تدرج الرمادي لتهيئتها لعملية كشف الوجوه.

كشف الوجوه باستخدام OpenCV

يتم استخدام Haar Cascade لاكتشاف مواقع الوجوه داخل الصورة وتحديد أبعاد كل وجه.

التعرّف على الوجوه

تتم مقارنة الوجوه المكتشفة مع الوجوه المخزنة في قاعدة البيانات باستخدام أسلوب Template Matching لتحديد الهوية الأقرب.

تخزين نتائج التعرّف

تُسجّل بيانات كل عملية تعرّف في قاعدة البيانات، متضمنة اسم الشخص، موقع الوجه، وتاريخ ووقت الاكتشاف.

إنشاء التقارير

يتم إنشاء تقرير بصيغة PDF لكل عملية تعرّف، يحتوي على صورة الوجه وبيانات الشخص ووقت تسجيل الدخول.

عرض النتائج للمستخدم

تُعرض الصورة النهائية مع تحديد الوجوه وأسمائها، ثم تُحفظ جميع التغييرات وتُغلق قاعدة البيانات.

بناء المشروع

1 بناء قاعدة البيانات.

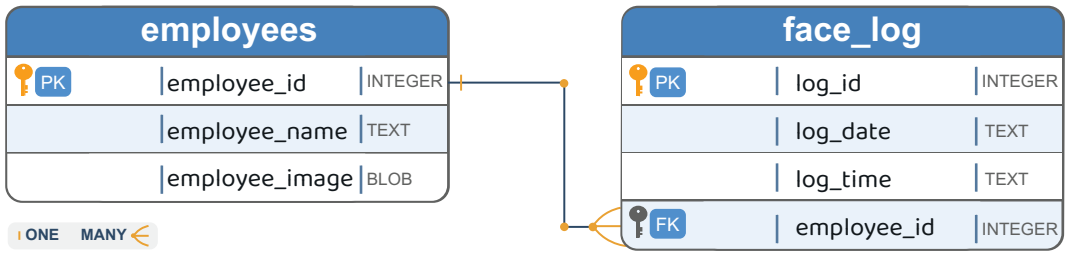
تعتمد قاعدة البيانات في هذا المشروع على عدة جداول مترابطة، صُمِّمت لتنظيم بيانات الوجوه والمستخدمين وتسجيل نتائج عمليات التعرّف بشكل دقيق ومنهجي.

employees: يُستخدم لتخزين بيانات الحسابات، ويظهر في التعليمات البرمجية كجدول الأب Parent يُخزّن صور الوجوه داخل قاعدة البيانات بصيغة ثنائية (BLOB)، ويُستخدم مباشرة في عملية التعرّف على الوجوه، حيث يتم استرجاع الصورة وتحويلها إلى مصفوفة NumPy للمقارنة، حيث يُمثّل المرجع الرئيسي للحساب (employee_id).

```
cursor.execute('''
    CREATE TABLE IF NOT EXISTS employees (
        employee_id INTEGER PRIMARY KEY AUTOINCREMENT,
        employee_name TEXT UNIQUE,
        employee_image BLOB
    )
''')
```

face_log: يُستخدم لتسجيل نتائج كل عملية تسجيل دخول، ويتضمن تاريخ ووقت تسجيل الدخول، ورقم الموظف employee_id، ويُستخدم لأغراض التوثيق وإعداد التقارير.

```
cursor.execute('''
    CREATE TABLE IF NOT EXISTS face_log (
        log_id INTEGER PRIMARY KEY AUTOINCREMENT,
        employee_id INTEGER,
        log_date TEXT,
        log_time TEXT,
        FOREIGN KEY(employee_id) REFERENCES employees(employee_id)
    )
''')
```



شكل (7-1) يُمثل مخطط ERD لقاعدة بيانات تسجيل دخول الموظفين.

2 تعريف دالة التعرف على الوجوه.

def recognize_face_by_template(face_gray):

- تقارن هذه الدالة الوجه المقتطع من الصورة بالصور المحفوظة في قاعدة البيانات.
- تستخدم خوارزمية matchTemplate من OpenCV التي تقيس درجة التشابه بين صورتين.

منطق المقارنة:

result = cv2.matchTemplate(known_image, face_resized, cv2.TM_CCOEFF_NORMED)

- score يمثل نسبة التشابه.
- إذا تجاوزت النسبة 0.3 أو 0.7، تعتبر مطابقة مقبولة.

3 إنشاء اتصال مع قاعدة البيانات.

```
connection = sqlite3.connect('face_detections.db')

cursor = connection.cursor()
```

◄ يتم إنشاء اتصال بقاعدة بيانات face_detections.db.

4 قراءة الصورة من المستخدم.

```
image_path = input('Enter the image path (e.g., 1.png): ').strip()

image = cv2.imread(image_path)
```

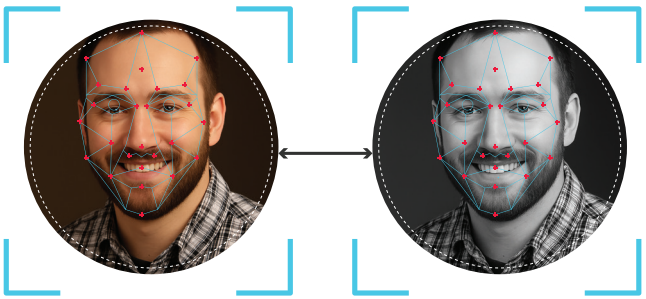
◄ يحصل البرنامج على مسار الصورة من المستخدم.

◄ يتحقق من إمكانية تحميل الصورة.

5 تحويل الصورة إلى تدرج رمادي.

```
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

◄ التعرف على الوجه يتم بشكل أكثر كفاءة باستخدام صور رمادية بدلًا من الألوان.



6 كشف الوجوه باستخدام كاشف Haar.

```
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_
frontalface_default.xml')
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=50)
```

- يستخدم كاشف جاهز من حزمة OpenCV.
- يعيد مجموعة إحداثيات (x, y, w, h) لكل وجه مكتشف.

7 التعامل مع كل وجه مكتشف.

لكل وجه:

- يُقص الوجه من الصورة.
- يُستخدم في محاولة التعرف على وجه محدد recognize_face_by_template.
- تُرسم مستطيلات وتوضع أسماء على الصورة.

```
cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)

cv2.putText(image, name, (x, y + h + 25), ...)
```

8 حفظ النتائج.

```
cursor.execute('''
    INSERT INTO face_log (log_time, log_date, employee_id)
    VALUES (?, ?, (SELECT employee_id FROM employees WHERE employee_name = ?))
''', (register_time, register_date, name))
```

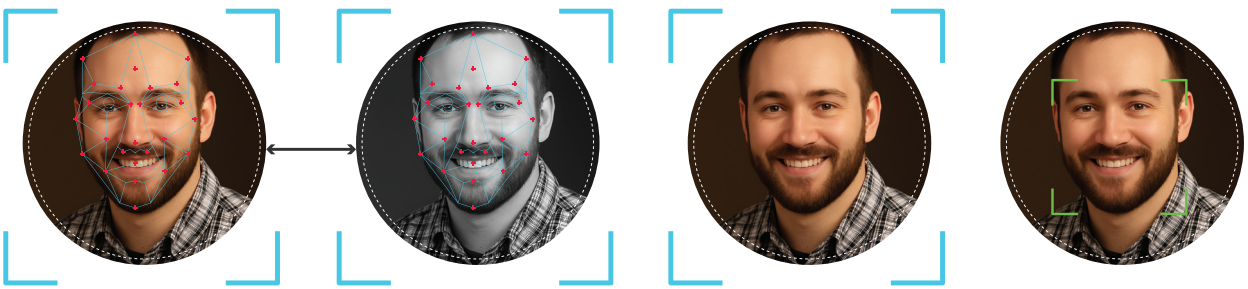
◄ يتم تسجيل كل عملية تعرف على الوجه مع الوقت والتاريخ في قاعدة البيانات.

9 عرض الصورة المُعدلة.

```
cv2.imshow('Identified Faces', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

◄ تعرض الصورة بعد تمييز الوجوه والتعرف عليها.

◄ إغلاق جميع النوافذ المفتوحة التي تم فتحها باستخدام دالة cv2.imshow بالضغط على أي مفتاح من لوحة المفاتيح.



10 إنشاء التقارير.

يُمكن إنشاء التقارير برمجيًا باستخدام حزمة ReportLab التي تتيح إنتاج ملفات PDF، وإمكانية إضافة النصوص والصور والعناوين وتنسيق الصفحات، مما يجعلها مناسبة لإعداد التقارير بصورة منظمة واحترافية.

def create_report():

إنشاء تقرير PDF باستخدام ReportLab، يتضمن اسم وصورة الموظف، وتاريخ ووقت التعرّف على الوجه من قاعدة البيانات.

no.	Name	Time	Date	Image
1	Fahed	20:32:41	2025-12-20	
2	Fahed	20:33:23	2025-12-21	
3	Fahed	20:36:22	2025-12-22	
4	Fahed	23:56:38	2025-12-23	

شكل (7-1) يُمثل تقرير بيانات تسجيل دخول الموظفين.

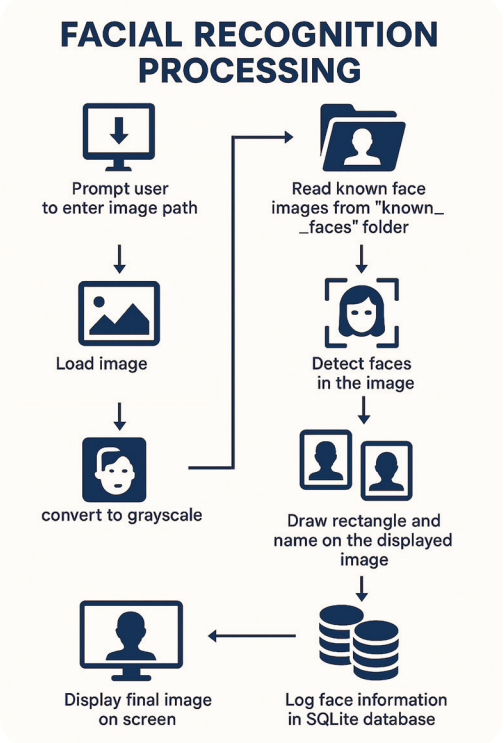
التعليمات البرمجية المقترحة لتنفيذ المشروع

```
import cv2
import os
import sqlite3
from datetime import datetime

# Function to recognize a face using template matching
def recognize_face_by_template(face_gray):
  best_match_name = "Unknown"
  highest_score = 0

  for filename in os.listdir("known_faces"):
    print(f"Comparing with image: {filename}")
    known_image = cv2.imread(f"known_faces/{filename}", 0)
    if known_image is None:
      print(f"Failed to load image: {filename}")
      continue

    try:
      face_resized = cv2.resize(face_gray, known_image.shape[::-1])
    except Exception as e:
      print(f"Failed to resize face for comparison with {filename}: {e}")
      continue

    result = cv2.matchTemplate(known_image, face_resized, cv2.TM_CCOEFF_NORMED)
    _, score, _, _ = cv2.minMaxLoc(result)
    print(f"Match score with {filename}: {score:.2f}")

    if score > highest_score and score > 0.3:
      highest_score = score
      best_match_name = os.path.splitext(filename)[0]

  return best_match_name

# Create or connect to the database
connection = sqlite3.connect("face_detections.db")
cursor = connection.cursor()
cursor.execute("""
  CREATE TABLE IF NOT EXISTS face_log (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    image_name TEXT,
    person_name TEXT,
    x INTEGER,
    y INTEGER,
    w INTEGER,
    h INTEGER,
    detected_at TEXT
```

```
    )
""")

# Request image path from user
image_path = input("Enter image path (e.g., test.jpg or C:/Users/...): ").strip()
image = cv2.imread(image_path)

if image is None:
    print("Failed to load image. Please check the path.")
    exit()

gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Detect faces
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5)

print(f"Number of detected faces: {len(faces)}")

# Process each detected face
for (x, y, w, h) in faces:
    face_crop = gray[y:y+h, x:x+w]
    name = recognize_face_by_template(face_crop)
    print(f"Detected face at ({x},{y}) recognized as: {name}")

    # Draw rectangle and label on the image
    cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
    cv2.putText(image, name, (x, y + h + 25), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 0), 2)

    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    cursor.execute("""
        INSERT INTO face_log (image_name, person_name, x, y, w, h, detected_at)
        VALUES (?, ?, ?, ?, ?, ?, ?)
    """, (os.path.basename(image_path), name, x, y, w, h, timestamp))

# Commit and close the database
connection.commit()
connection.close()

print("Face data saved to database.")
cv2.imshow("Identified Faces", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

ما الذي يحدث عند تشغيل البرنامج

1. طلب إدخال مسار الصورة من المستخدم لمعالجتها.
2. تحميل الصورة المحددة وتحويلها إلى تدرج رمادي Grayscale لتسهيل المعالجة.
3. اكتشاف الوجوه في الصورة باستخدام خوارزمية Haar Cascade من OpenCV.
4. مقارنة كل وجه مُكتشف مع الصور من قاعدة البيانات باستخدام template matching لتحديد هوية الشخص.
5. رسم مستطيل حول الوجه وكتابة الاسم عليه داخل الصورة المعروضة.
6. تسجيل رقم الموظف ووقت وتاريخ الدخول في قاعدة بيانات.
7. عرض الصورة النهائية على الشاشة مع التعديلات.



8. إنشاء تقرير بعمليات الدخول للموظف من قاعدة البيانات بناءً على طلب المُستخدم.

معاينة قاعدة البيانات

يُمكن معاينة هيكلية قاعدة البيانات face_detection.db وبياناتها، من خلال برنامج DB Browser for SQLite.

التقنيات المُستخدمة في المشروع

1 الفرق بين تقنية كشف الوجه وتقنية التعرف عليه.

كشف الوجه Face Detection:
عملية تحديد موقع وجود وجه داخل صورة أو فيديو، أي تحديد الإحداثيات (x, y)، والأبعاد الطول والعرض (h, w).

التعرف على الوجه Face Recognition:
عملية مقارنة الوجه المكتشف مع مجموعة من الوجوه المعروفة لتحديد هوية الشخص.

2 استخدامات حزمة OpenCV / CV2 في هذا المشروع.

- قراءة الصور.
- تحويل الصور إلى رمادي.
- كشف الوجوه باستخدام كواشف مثل Haar Cascade.
- مطابقة الوجوه بواسطة matchTemplate.
- رسم المستطيلات وكتابة النصوص على الصور.
- عرض الصورة النهائية.

3 تحويل الصورة إلى تدرج رمادي.

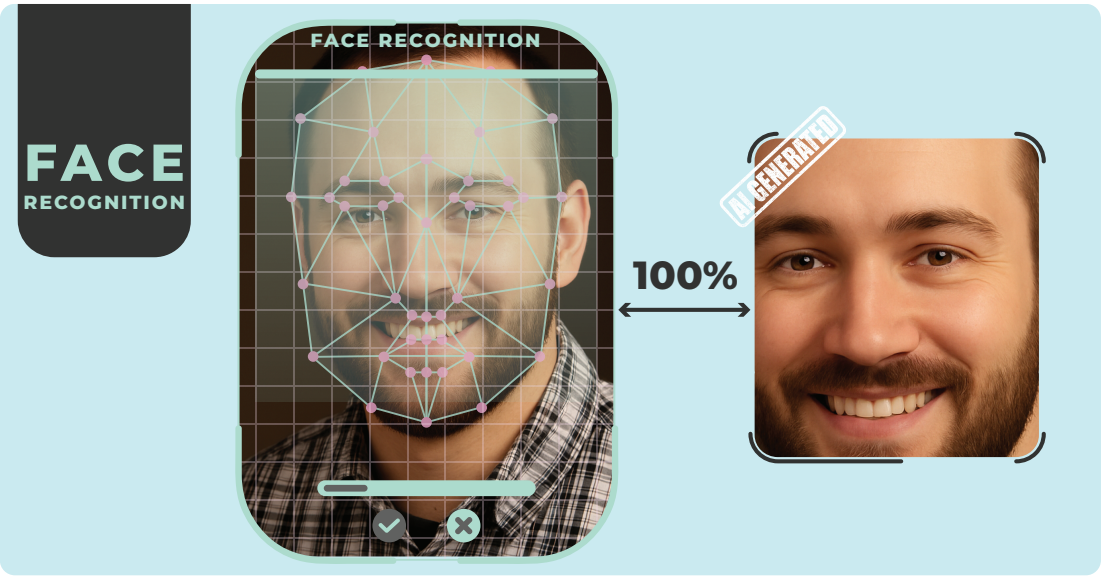
- يتم تحويل الصورة إلى تدرج رمادي حيث خوارزميات كشف الوجوه والتعرف عليها تعمل بشكل أسرع وأكفأ على الصور الرمادية (ذات القناة اللونية الواحدة) بدلاً من الصور الملونة (ذات 3 قنوات).
- تقليل حجم البيانات المراد معالجتها.
- بعض التقنيات تعتمد على تباين الإضاءة وليس الألوان.

4 استخدام قاعدة بيانات بدلًا من ملف نصي.

- تنظيم البيانات: تسهل التعامل مع بيانات كثيرة ومنظمة (مثل اسم الوجه، الوقت، المكان).
- البحث والاسترجاع السريع: يمكن استخدام استعلامات SQL للبحث والتحليل بسهولة.
- عدم التكرار: يمكن ضبط عبر القيود في الجداول.
- الاعتمادية والأمان: أكثر أمانًا ودقة في الحفظ والاسترجاع مقارنة بملف نصي.

5 استخدام matchTemplate.

- matchTemplate هي دالة من OpenCV تقوم بمقارنة صورة صغيرة (الوجه المُكتشف) مع صورة كبيرة (الوجه المعروف).
- تعيد (درجة التشابه) بين الصورتين كنسبة، ومن خلالها نحدد إذا كان الوجه مطابقًا أم لا.



المنتجات الرقمية

مراحل تصميم وتطوير المنتج الرقمي

Designing and Developing a Digital Product

نواتج التعـلم

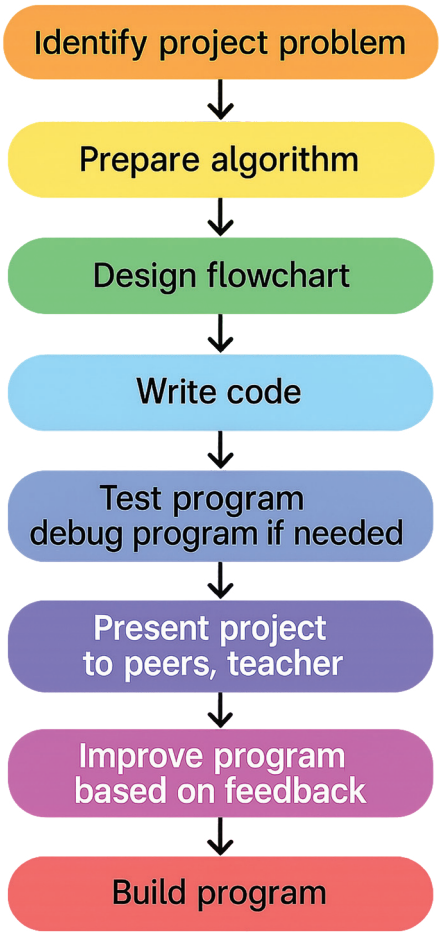
- توظيف مهارات التصميم والبرمجة لحل مشكلة واقعية بطريقة إبداعية.
- التعاون ضمن فريق لتوزيع المهام وتنفيذ المشروع حسب خطة مُنظمة.
- توثيق مراحل المشروع، وعرضه على المعلم والزملاء مع التغذية الراجعة.

يمثـل رمـز الاسـتجابة السـريعة QR رابـط
لملفـات أوراق العمـل، ومصـادر التعلـم

وحدة المشروعات

إن الهدف الأساسي من إنتاج المشروع هو الاستفادة من المهارات التي تم دراستها وتطبيقها من خلال أوراق العمل بالإضافة إلى تنمية مهارات العمل الجماعي التعاوني، والقدرة على تجميع الوسائط اللازمة لإنتاج المشروع وفق أُسس تحليل النظم والبرمجة والقدرة على تنمية الابتكار.

من خلال دراستك لوحدة البرمجة بلغة Python، والاطلاع على تطبيقاتها المتعددة، والمساعدة المتوافرة في رمز الاستجابة السريع QR في بداية وحدة المُنتجات الرقمية، صمم المشروع الخاص بك، متبعًا الخطوات التالية:



- تحديد مشكلة المشروع.
- إعداد الخوارزمية.
- تصميم خريطة التدفق.
- كتابة التعليمات البرمجية.
- اختبار البرنامج، تصحيح البرنامج إن وجدت.
- عرض المشروع على زملائك، ومعلمك.
- تحسين البرنامج وفق التغذية الراجعة.
- بناء البرنامج.

شكل (8-1) يُمثل خطوات تصميم المشروع.

ضوابط وإرشادات تنفيذ المشروع

سنستعرض الآن طريقة إعداد المشاريع ضمن ضوابط وإرشادات محددة لتنسيق العمل باتباع الخطوات التالية:

مخطط إعداد مشروع برمجي

شكل (8-2) يُمثل مخطط إعداد مشروع برمجي.

مهام فريق العمل

يتكون الفريق من 4 إلى 5 أعضاء، تُقسم المهام بينهم على النحو التالي:

- قائد الفريق: المسؤول عن إدارة الفريق وتجميع الملفات والمستندات المطلوبة وتقديمها للمعلم.
- مُعد الخوارزمية: المسؤول عن إعداد الخطوات الأساسية لعمل المشروع.
- مصمم خريطة التدفق: المسؤول عن ترجمة الخوارزمية إلى خريطة التدفق.
- المبرمج: المسؤول عن ترجمة خريطة التدفق لتعليمات برمجية واختبارها والتأكد من سلامتها.
- مُعد العرض التقديمي: المسؤول عن إعداد وتصميم العرض التقديمي للمناقشة.

تُقسم المهام بين أعضاء الفريق على النحو التالي:

اسم المتعلم	المهمة	الوظيفة
	قائد الفريق	المسؤول عن إدارة عمل الفريق وتجميع الملفات والمستندات المطلوبة وتقديمها للمعلم.

جدول (8-1) يُمثل تقسيم المهام بين أعضاء الفريق.

مراحل المشروع

الأسبوع	المرحلة	الناتج المُتوقع
الأول	التخطيط والتحليل	اختيار فكرة المشروع.
		تحديد المشكلة والأهداف.
		إعداد وثيقة المتطلبات الأولية.
الثاني	خطة المشروع والتوزيع	توزيع المهام على أعضاء الفريق.
		إعداد خطة تنفيذ المشروع أسبوعياً.
الثالث	التصميم والتوثيق الأولي	تصميم الواجهة.
		إعداد خطة العمل بالتفصيل.
الرابع	البرمجة والتنفيذ الأولي	البدء في تنفيذ المشروع عمليًا.
		تجربة الأدوات البرمجية والتقنية.
الخامس	الاختبار والتحسين	مراجعة الأخطاء البرمجية.
		اختبار الأداء.
		إجراء التعديلات والتحسينات النهائية.
السادس	**التوثيق النهائي والعرض**	كتابة التقرير النهائي.
		إعداد العرض التقديمي.
		التدريب على العرض وتقديمه أمام الصف.

جدول (8-2) يُمثل المراحل الأساسية للمشروع.

ملاحظة: يختلف عدد أسابيع المشروع والمراحل المختلفة وفق خطة توزيع المنهج.

فكرة المشروع

تصميم مشـــــروع تقني تعليمي بســـــيط لحل مشـــــكلة أو تنفيذ فكرة مفيدة بطريقـــــة إبداعية على أن تكون واضحـــــة وقابلة للتطبيق.

يناقــش أعضــاء الفريــق ويحــدد موضــوع يثيــر اهتمامهــم، ســواءً كان تطبيقــاً أو لعبــة أو أداة تحليــل البيانــات وغيرهــا، علــى أن تكــون مُطابقــة للشــروط التاليــة:

- أن تكون الفكرة واضحة وقابلة للتطبيق مع شرح المشكلة والحلول.
- تحديد الفئة المستفادة.
- أن تكون الفكرة قابلة لتصميم نموذج برمجي.
- أن تحتوي على عدة مصادر خارجية يسهل الرجوع إليها.
- أن تعتمد على تقنية اكتشاف الوجوه أو أي تقنية من تقنيات الذكاء الاصطناعي.

سجل فكرة المشروع بعد مناقشة أعضاء الفريق:

أهداف المشروع

يهدف المشروع إلى إكساب المتعلم مهارات تقنية وإبداعية عبر تصميم وتنفيذ فكرة تعليمية بطريقة منظمة وتعاونية.

سجل أهداف مشروعك:

التوظيف الواقعي في الحياة العملية

يساهم المشروع في ربط التقنية بواقع الحياة من خلال توظيف البرمجة والذكاء الاصطناعي وقواعد البيانات في حل مشكلات حقيقية أو تحسين ممارسات يومية بطريقة ابتكارية.

سجل كيف ستوظف المشروع الخاص بك في الحياة العملية:

مهارات المشروع

يهدف المشروع إلى تنمية مهارات المتعلمين التقنية مثل التصميم والبرمجة، إلى جانب المهارات الشخصية كالتعاون وحل المشكلات، وذلك من خلال تنفيذ فكرة عملية لإكسابهم خبرات واقعية تعزز جاهزيتهم للمستقبل، وتساعدهم على اكتساب كفاءات تقنية تعزز قدرتهم على الابتكار والتطبيق العملي.

حدد المهارات المستخدمة بالمشروع:

	المجال	المهارات الفرعية المستهدفة
	المهارات الرقمية والتقنية	القدرة على كتابة أو تعديل أكواد برمجية بلغة Python.
		استخدام التقنيات المناسبة لقراءة الوجوه والتعرف عليها تلقائيًا.
		تخزين ومعالجة البيانات باستخدام قاعدة بيانات SQLite.
		معرفة كيفية حفظ الصور والتعامل مع ملفات الوجوه المخزنة.
	التفكير الناقد وحل المشكلات	تحليل المشكلة وتحديد عناصرها.
		تجريب الحلول واختيار الأنسب.
		التحقق من دقة النتائج وتعديل الأخطاء.
	مهارات التوثيق والبحث العلمي	تصوير مراحل تنفيذ المشروع وتسجيلها.
		جمع معلومات من مصادر موثوقة.
		كتابة المراجع العلمية.
	مهارات العمل الجماعي	التعاون وتقسيم المهام.
		تحمل المسؤولية والالتزام بالوقت.
		اتخاذ قرارات جماعية.
	مهارات العرض والتواصل	إعداد عرض بصري فعّال (شرائح، فيديو، أدلة، تقارير).
		شرح الفكرة بلغة واضحة ومقنعة.
		التفاعل مع الجمهور والرد على الأسئلة.

جدول (8-3) يُمثل مهارات المشروع الأساسية.

خطوات تنفيذ المشروع

1. إعداد بيئة العمل

تثبيت ما يحتاج إليه الفريق من برمجيات وأدوات لإعداد المشروع، ومنها:

- PyCharm: لتصميم النموذج البرمجي واستيراد المكاتب اللازمة.
- PowerPoint: لإعداد العرض التقديمي.
- Windows screen recorder: لتسجيل الشاشة إستعداداً لتوثيق المشروع.
- Draw.io: لرسم خريطة التدفق للمشروع.
- Microsoft Edge: للبحث عن المعلومات.

و أي مستلزمات أخرى تخدم المشروع.

2. إعداد هيكل المشروع

يُفضل الاستغلال الأمثل للمهارات البرمجية التالية:

- استخدام لغة Python.
- إنشاء قاعدة بيانات تحتوي على جدولين على الأقل بينهما علاقة.
- استخدام العمليات الأساسية CRUD.
- معالجة الأخطاء باستخدام الاستثناءات try/ except.

3. كتابة التعليمات البرمجية

يكتب المبرمج التعليمات البرمجية استناداً لما تم تخطيطه سابقاً.

4. اختبار المشروع

يختبر المبرمج البرنامج، ويتأكد من خلوه من الأخطاء البرمجية، اللغوية، والعلمية.

5. تطوير المشروع

يُبسط المبرمج ويرتب التعليمات البرمجية إذا أصبحت صعبة القراءة، وإضافة التعليقات والبيانات الإرشادية التي توضح مهمة التعليمات البرمجية.

6. توثيق المشروع

يُنشئ مصمم العرض التقديمي عرضًا شاملاً للجوانب التالية:

- عرض بيانات المدرسة وأعضاء الفريق.
- الترحيب بالمعلم وزملائه المتعلمين.
- عرض مقدمة عن فكرة المشروع موضحاً الهدف والمشكلة والفئة المستفادة والحل.
- عرض خريطة التدفق.
- عرض النموذج البرمجي.
- عرض المصادر.
- فتح باب الأسئلة والمناقشة.
- الختام.

و تسليم مجلد مجمع لجميع ملفات المشروع.

7. مشاركة المشروع

عرض الفريق للمشروع، ومناقشته مع المعلم والمتعلمين.

8. الاستمرار في التعلم

التوسع في الاطلاع على مستجدات الذكاء الاصطناعي وتنمية قدرات المتعلم البرمجية من خلال الدورات التدريبية والمنتديات الحاسوبية.

التوثيق العلمي للمشروع

- المراجع:
  - الالتزام بطريقة علمية لتوثيق المراجع على سبيل المثال نظام APA[1].
  - عرض ومشاركة المشروع.
- يقوم الفريق بعرض المشروع ومناقشته مع المعلم والمتعلمين، مع مشاركته على منصة Teams.
- الاستمرار في التعلم.

مشاريع إثرائية

1 اختصار لعبارة American Psychological Association

ويُعد أحد أشهر أنظمة التوثيق المرجعي المستخدمة عالميًا، خاصةً في الأبحاث العلمية، والعلوم الاجتماعية، والنفسية، والتربوية، والبحث الأكاديمي عمومًا.

المراجع

- مؤسسة بايثون للبرمجيات. (2024). دليل بايثون الرسمي: إصدار 7. https://www.python.org/doc.
- منظمة الأمم المتحدة للتربية والعلم والثقافة، « الذكاء الاصطناعي في التعليم» (2017).
- خارطة الطريق للتحول الرقمي للمؤسسات الحكومية في دولة الكويت، د. عبد الله محمد عبد الكريم المطوع، مركز دراسات الخليج والجزيرة العربية. العدد 32 . 2023م.
- التوصية الخاصة بأخلاقيات الذكاء الاصطناعي- منظمة الأمم المتحدة للتربية والعلم والثقافة (اليونسكو)، 2021م.
- محمد لحلح. (2020). مدخل الى الذكاء الاصطناعي وتعلم الآلة. شركة حسوب وأكاديمية حسوب.
- نرمين مجدي. (2020). الذكاء الاصطناعي وتعلم الآلة. صندوق النقد العربي.
- منظمة الأمم المتحدة للتربية والعلم والثقافة. (2021). الذكاء الاصطناعي والتعليم. منظمة الأمم المتحدة للتربية والعلم والثقافة.
- سدايا. (سبتمبر 2023). مبادئ أخلاقيات الذكاء الاصطناعي. سدايا.
- Basu, S. (2021, May 20). Learn SQLite with Python in 24 Hours: Simple, Concise & Easy Guide to Using Database with Python. Independently published.
- Kurniawan, A. (2021, January 30). Python and SQLite Development. PE Press.
- Siahaan, V. S., & Sianipar, R. H. (2025). Learn SQLite with Python: Building Database-Driven DesKtop Projects. Independently published.
- Hayes, W. (2025, [Feb]). Master Computer Vision and AI with OpenCV and Python: UnlocK Cutting-Edge Tools, Techniques, and Real-World Applications for Image Processing and Machine Learning. Independently published.
- Howse, J., & Minichino, J. (2020, Feb 20). Learning OpenCV 4 Computer Vision with Python 3 (3rd ed.). PacKt Publishing.
- Mugesh, S. (2023). Hands-on ML Projects with OpenCV: Master computer vision and Machine Learning using OpenCV and Python. Independently published.
- Diagram illustrating SQL join types (INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN, CROSS JOIN). (n.d.). Byte-ByteGo. Retrieved from https://bytebytego.com.

11