قواعد معطيات متقدمة

معالجة الاستعلام Query processing

مخطط الفصل

- الخطوات الأساسية في معالجة الاستعلام
- قياس كلفة الاستعلام
- تعليمة الاختيار
- الفرز
- الدمج
- عمليات أخرى
- تقييم التعابير

تنفيذ اختيارات معقدة

- **اختيار الـ (Conjunction (and** $\sigma_{\theta 1 \wedge \theta 2 \wedge \ldots \theta n}(r)$
- *حساب حجم النتيجة المتوقعة من الاختيار السابق :*

– بفرض أن عدد التسجيلات الناتجة من $\sigma_{\theta i}(r)$ هو S_i فيكون احتمال أن يحقق سطر من العلاقة الشرط هو S_i / n_r حيث n_r عدد أسطر العلاقة

– بفرض أن الأسطر المحققة لكل شرط على حدى مستقلة عن بعضها وبالتالي يكون احتمال أن يحقق سطر من العلاقة الشرط $\sigma_{\theta 1 \wedge \theta 2 \wedge \ldots \theta n}(r)$ هو $s_1 * s_2 * \ldots . * s_n / n_r^n$

– عدد الأسطر المتوقع التي تحقق الشرط هو

$$Sc(A, r) = n_r * s_1 * s_2 * \ldots . * s_n / n_r^n$$

خوارزميات الاختيارات المعقدة

- اختيار **Disjunction (or)** $\sigma_{\theta 1 \vee \theta 2 \vee \ldots \theta n}(r)$:
- **A11** (*disjunctive selection by union of identifiers*).
  - قابل للتطبيق في حال كان هناك فهرس لجميع الشروط
    - في الحالات الأخرى يُستخدم المسح خطياً
  - اجتماع المؤشرات الناتجة عن كل شرط
  - إظهار النتيجة من الملف
- النفي **Negation** $\sigma_{\neg \theta}(r)$:
  - استخدام المسح الخطي للملف
  - في حال كان عدد أسطر العلاقة المحققة للشرط $\theta_{\neg}$ قليل جداً،
    - يُستخدم الفهرس إن وجد على θ لإيجاد مؤشرات نحو التسجيلات غير المحققة للشرط ومن ثم إظهارها من الملف مباشرة.

الفرز

- ✓ عملية π دوماً تحتاج لعملية فرز وذلك لحذف الأسطر المكررة ، أيضاً الدوال التجميعية (count , sum , ...) تحتاج لفرز لكي تظهر النتائج وراء بعضها ..
- يمكننا بناء فهرس للعلاقة واستخدامه لقراءة العلاقة بترتيب مفروز، يقود ذلك إلى وصول كتلي من أجل كل سطر من العلاقة
- من أجل العلاقات التي يمكن شحنها بشكل كامل في الذاكرة،
  - يمكن استخدام خوارزميات quicksort الفرز السريع
- من أجل العلاقات التي لا يمكن شحنها بشكل كامل في الذاكرة
  - يمكن اختيار خوارزمية الفرز بالدمج الخارجي **external sort-merge**

الفرز بالدمج الخارجي- external sort-merge

لتكن M حجم الذاكرة المتاح لعملية الفرز (عدد الصفحات أو الكتل)، نحتاج إلى عدة أطوار لإنجاز عملية الفرز. يتألف كل طور من مرحلتين :

١. **مرحلة التجزئة Create sorted runs : تجزئة العلاقة إلى مجموعة من العلاقات الجزئية التي يمكن فرزها في الذاكرة المركزية المتاحة**
   - **نقوم بتكرار مجموعة من العمليات حتى نهاية العلاقة :**
     - قراءة M *كتلة من العلاقة* وشحنها *في الذاكرة*
     - الفرز في الذاكرة
     - كتابة النتيجة في run العلاقة الجزئية Ri
     - i:= i+1

ليكن N عدد مسارات التجزئة (عدد العلاقات الجزئية الناتجة لدينا من عملية التجزئة)،

٢. **مرحلة الدمج : دمج العلاقات الجزئية الناتجة (باستخدام N-way للدمج). وبفرض أن N < M**
   ١. **نستخدم N كتلة من الذاكرة (buffer) كدخل من العلاقات الجزئية، وكتلة من الذاكرة(buffer) كخرج. قراءة كتلة من كل علاقة جزئية ووضعها في الـbuffer**

repeat
- Select the first record (in sort order) among all buffer pages
- Write the record to the output buffer. If the output buffer is full write it to disk.
- Delete the record from its input buffer page.
  If the buffer page becomes empty **then**
  read the next block (if any) of the run into the buffer.

until all input buffer pages are empty:

في حال كانت M<= N نحتاج إلى عدة أطوار من الدمج، في كل طور يجري دمج M-1 علاقة جزئية وبالتالي ينقص عدد العلاقات الجزئية بنسبة M-1 في كل طور.

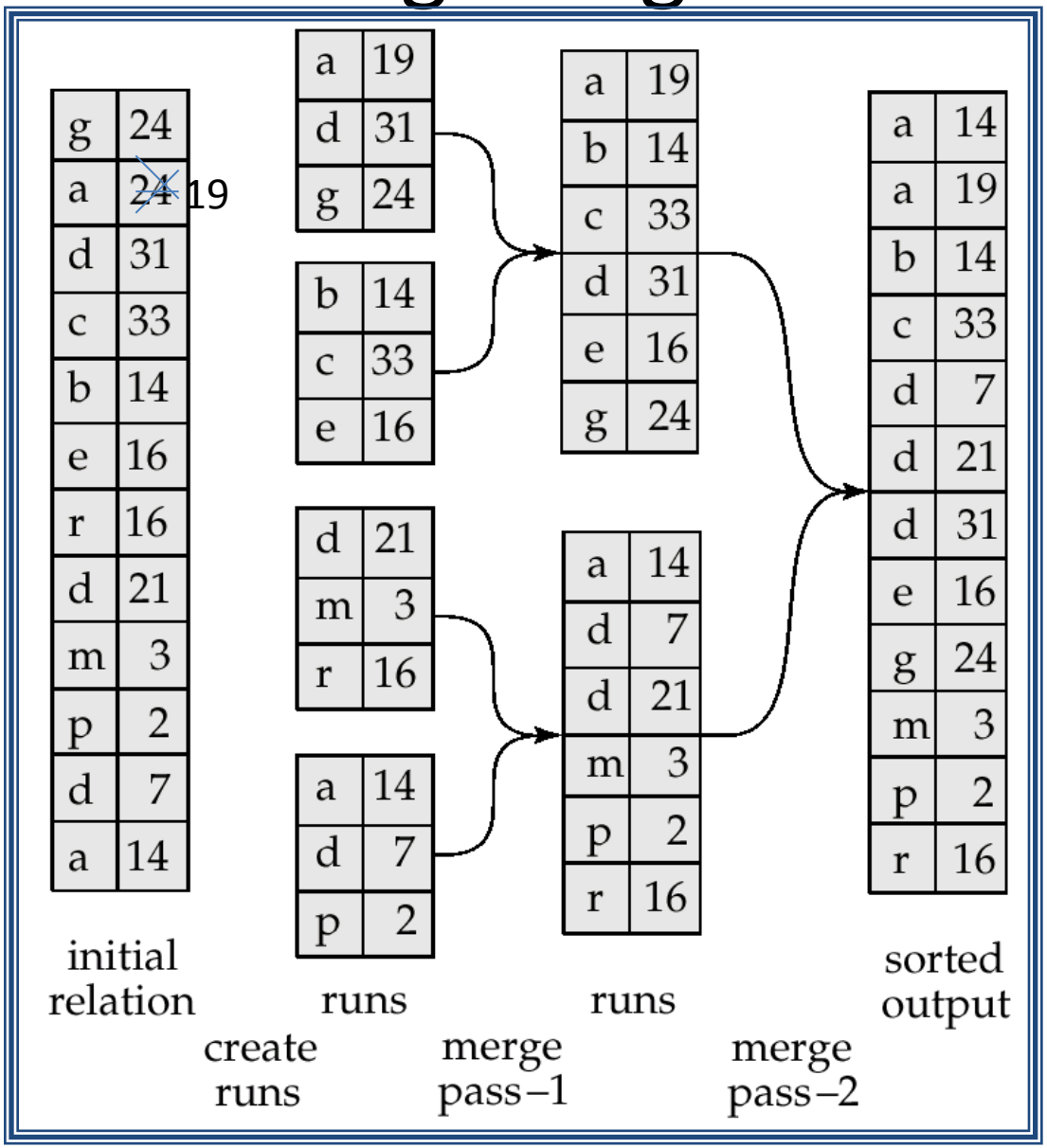
External sorting using sort-merge :مثال

بفرض أن M=3 Blocks (أي أستطيع شحن ٣ سجلات)

و حجم الكتلة = 1تسجيلة و br=12

g	24
a	١٩
d	31
c	33
b	14
e	16
r	21
d	21
m	3
p	2
d	7
a	14

External sorting using sort-merge :مثال



Join operation – عملية الدمج

- يوجد عدة طرق لتنفيذ عملية الدمج
  - الدمج بحلقة متداخلة على مستوى السطر Nested-loop join
  - الدمج بحلقة متداخلة على مستوى الكتلة Block nested-loop join
  - الدمج بحلقة متداخلة باستخدام الفهرس Indexed nested-loop join
  - الدمج الممزوج Merge-join
  - الدمج باستخدام التقطيع Hash-join
- اختيار الطريقة اعتماداً على الكلفة المتوقعة
- يمكن تقدير حجم العلاقة الناتجة من عملية الدمج بالشكل التالي :
  - إذا كان $R \cap S = \emptyset$ فعملية الدمج مطابقة لعملية الجداء الديكارتي
    - بالتالي يقدر عدد الأسطر بـ $n_r * n_s$
  - إذا كان $R \cap S = \{A\}$ يكون عدد الأسطر في العلاقة الناتجة عن الدمج
    - $n_r * n_s / V(A,r)$ أو $n_r * n_s / V(A,s)$

الدمج باستخدام حلقة متداخلة على مستوى السطر
Nested-loop join

- بمشي سطر سطر ، كل سطر بختبرو مع كل الأسطر تبع العلاقة التانية وبشوفو بيندمج ولا لأ .. إذا أي بيطلع output وإذا لأ بترك وبمشي عالسطر التاني .
- لحساب عملية الدمج وفق θ أي $r \bowtie_\theta s$

for each tuple t_r **in** r **do begin**
  for each tuple t_s **in** s **do begin**
    test pair (t_r, t_s) to see if they satisfy the join condition θ
    if they do, add $t_r \bullet t_s$ to the result.
  end
end

- r تدعى العلاقة الخارجية في عملية الدمج **outer relation** و s تدعى العلاقة الداخلية في عملية الدمج **inner relation**
- لا تتطلب وجود فهارس ويمكن استخدامها من أجل أي نوع من عمليات الدمج الشرطية
- مكلفة حيث أنها تختبر كل زوج من الأسطر في العلاقتين.

كلفة الدمج باستخدام حلقة متداخلة على مستوى السطر

- أسوأ حالة، إذا وجدت ذاكرة فقط للتعامل مع كتلة واحدة من كل علاقة، تكون الكلفة المتوقعة

$$n_r * b_s + b_r$$

- إذا كان من الممكن وضع العلاقة الصغيرة الحجم في الذاكرة، تُستخدم عندئذ كعلاقة داخلية مما يقلص الكلفة إلى $b_r + b_s$ وصول إلى القرص
- مثال توضيحي

من أجل قاعدة معطيات تحوي علاقة depositor بعدد أسطر يساوي ٥٠٠٠ سطر، سعة الكتلة ٥٠ تسجيلة، وعلاقة customer بعدد أسطر يصل إلى ١٠٠٠٠ سطر، سعة الكتلة ٢٥ تسجيلة.

- عدد الكتل اللازمة لتخزين علاقة depositor هو ١٠٠ كتلة، وعدد الكتل اللازمة لتخزين علاقة customer هو ٤٠٠ كتلة.
- وبفرض أنه لدينا أسوأ حالة بتوفر الذاكرة المتاحة
  - الكلفة المتوقعة إذا اعتبرنا علاقة depositor هي العلاقة الخارجية :

5000 * 400 + 100 = 2,000,100 block transfers

  - وإذا اعتبرنا علاقة customer هي العلاقة الخارجية :

10000 * 100 + 400 = 1,000,400 block transfers

- إذا وضعت العلاقة الصغيرة depositor بشكل كامل في الذاكرة، تصبح الكلفة المتوقعة

400+100=500 block transfers

الدمج بحلقة متداخلة على مستوى الكتلة
Block nested-loop join

- الاختلاف عن الدمج بحلقة متداخلة على مستوى السطر أنه في هذه الطريقة يجري اختبار كل كتلة من العلاقة الداخلية مع جميع كتل العلاقة الخارجية

```
for each block B_r of r do
    begin
        for each block B_s of s do
        begin
            for each tuple t_r in B_r do
            begin
                for each tuple t_s in B_s do
                begin
                    Check if (t_r, t_s) satisfy the join condition
                    if they do, add t_r • t_s to the result.
                end
            end
        end
    end
```

الدمج بحلقات متداخلة على مستوى الكتلة ـ تابع

- الكلفة المتوقعة : $b_r * b_s + b_r$ block transfers
- أفضل حالة : $b_r + b_s$ block transfers(وجود ذاكرة كافية لتخزين العلاقتين بشكل كامل)
- تحسينات على خوارزمية الدمج بحلقات متداخلة:
  - استخدام M-2 كتلة لشحن معطيات من العلاقة الخارجية، (M حجم الذاكرة المتاح لعملية الدمج). واستخدام الكتلتين الباقيتين لتخزين كتلة من العلاقة الداخلية وأخرى للعلاقة الناتجة.
    - الكلفة = $\lceil b_r / (M-2) \rceil * b_s + b_r$
  - في حال شكلت واصفة الدمج مفتاح في العلاقة الداخلية ، تتوقف الحلقة الداخلية حال الوصول إلى تطابق.
  - مسح العلاقة الداخلية من الاتجاهين للاستفادة من الكتل الموجودة في الذاكرة
  - استخدام الفهارس في حال وجودها.

الدمج بحلقات متداخلة باستخدام الفهرس
Indexed nested-loop join

- لاستعاضة عن المسح التسلسلي بمسح الفهرس في الحالات التالية :
  - الدمج هو دمج طبيعي (تساوي قيم واصفة الدمج)
  - وجود فهرس معرف على واصفة الدمج للعلاقة الداخلية (يمكن بناء فهرس فقط لحساب عملية الدمح)
- خوارزمية الحساب

For each tuple t_r in the outer relation *r*,
use the index to look up tuples in *s*
that satisfy the join condition with tuple t_r.

- أسوأ حالة :

توفر ذاكرة بسعة كتلة واحدة فقط من r ، ومن أجل كل سطر من r هناك بحث عن الأسطر المتوافقة معه باستخدام الفهرس.

كلفة الدمح = $b_r + n_r * c$

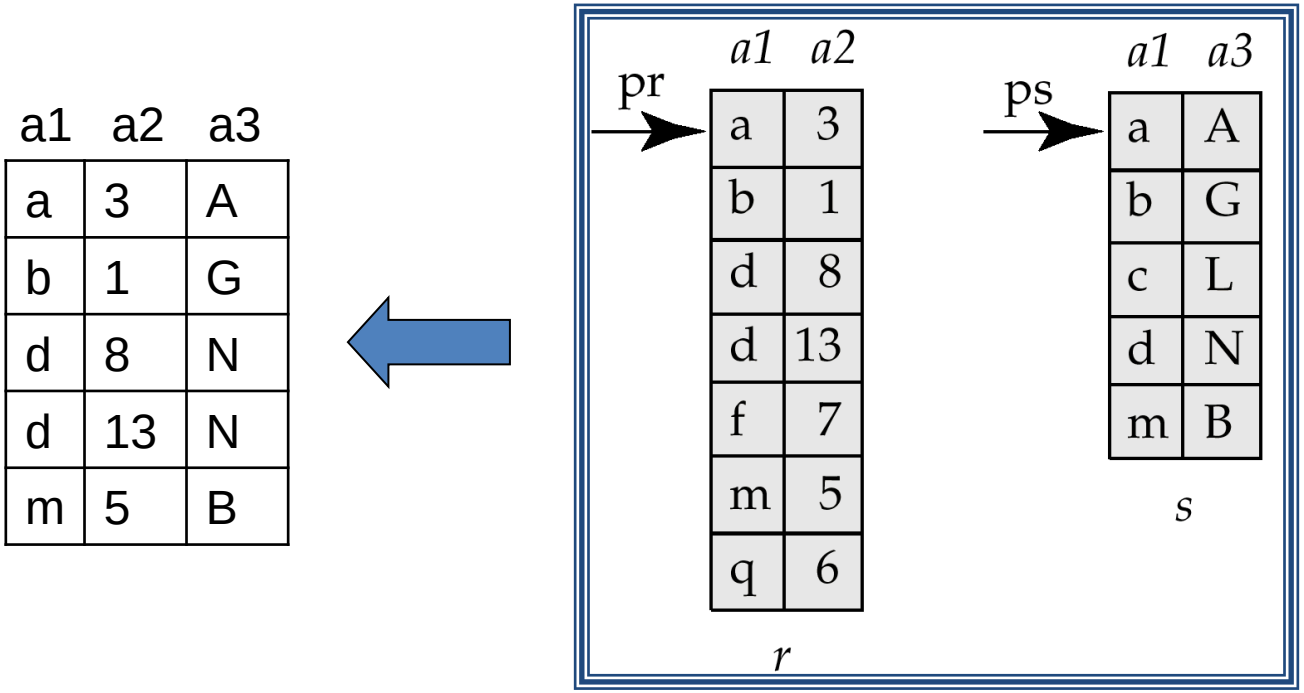
حيث c كلفة الوصول إلى الفهرس وإظهار الأسطر الموافقة.

- في حال توفرت فهارس للعلاقتين على واصفة الدمج r & s فمن المستحسن استخدام العلاقة ذات الأسطر الأقل كعلاقة خارجية.

Merge join

١- فرز العلاقتين بالنسبة لواصفة الدمج (في حال لم يكونا مفروزتين)

٢- عملية الدمج مشابهة لمرحلة الدمج في خوارزمية الفرز بالدمج، الخلاف بالتعامل مع القيم المكررة لواصفة الدمج حيث يتم احتساب كل زوج له نفس قيمة واصفة الدمج في النتيجة.



Joins with sorting

- Sort-Merge Join (conceptually)

(1) if R1 and R2 not sorted, sort them

(2) $i \leftarrow 1$; $j \leftarrow 1$;

While $(i \leq T(R1)) \wedge (j \leq T(R2))$ do

if R1[i].C = R2[j].C then OutputTuples

else if R1[i].C > R2[j].C then $j \leftarrow j+1$

else if R1[i].C < R2[j].C then $i \leftarrow i+1$

Procedure Output-Tuples

While (R1[i].C = R2[j].C) $\wedge$ (i $\leq$ T(R1)) do

{jj $\leftarrow$ j;

while (R1[i].C = R2[jj].C) $\wedge$ (jj $\leq$ T(R2)) do

{

output pair R1[i], R2[jj];

jj $\leftarrow$ jj+1

}

i $\leftarrow$ i+1 }

Merge join

- يستخدم فقط في الدمج المتساوي الطبيعي
- نحتاج إلى قراءة كل كتلة مرة واحدة (يمكننا ذلك من قراءة جميع أسطر العلاقتين)
- الكلفة تقدر :

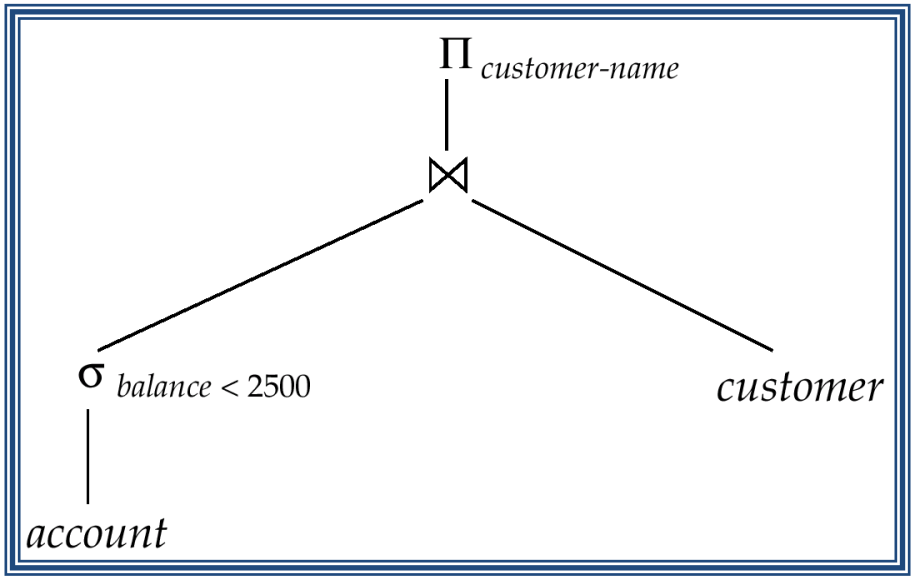
$b_r + b_s$ + the cost of sorting if relations are unsorted.

Evaluation of expressions تقييم التعابير

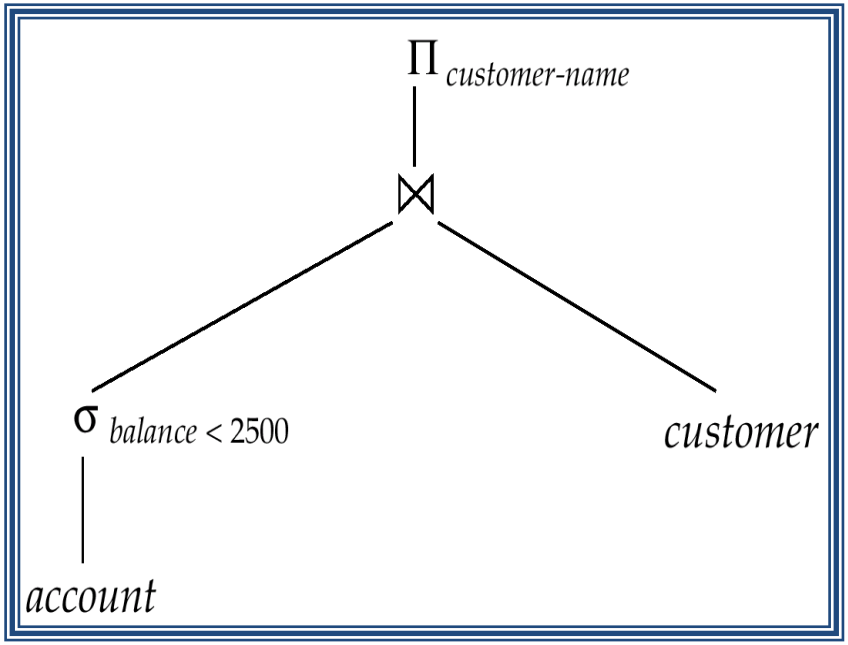
- **التجسيد Materialization**

– تنفيذ العمليات الجزئية على التتالي الواحدة تلو الأخرى، البدء من المستوى الأدنى ووضع النتائج الوسيطة في علاقات وسيطة لتدخل في تنفيذ العمليات التالية

- مثال : لإيجاد أسماء الزبائن الذين أرصدتهم أقل من ٢٥٠٠



Evaluation of expressions (cont.)



- **التنفيذ الأنبوبي Pipelining**
  - تنفيذ عدة عمليات بآن واحد بتمرير نتائج عملية مباشرة لإجراء العملية التالية عليها.
  - مثال : كما في المثال السابق فنتيجة عملية الاختيار لا يتم تخزينها
  - بدلاً من التخزين يجري تمرير الأسطر المحققة للشرط مباشرةً إلى عملية الدمج. وبشكل مشابه لايتم تخزين نتيجة عملية الدمج وإنما يتم تمريرها مباشرة إلى عملية الاسقاط.
- أقل كلفة من الطريقة السابقة materialization (لا حاجة لتخزين علاقات وسيطة على القرص)
- لا يمكن تطبيق هذه الطريقة دوماً . مثال : عملية الفرز، الدمج باستخدام التقطيع

عمليات الدمج المعقدة
Complex Joins (Cont.)

لحساب عملية دمج للعلاقات الثلاث $loan \bowtie depositor \bowtie customer$ *يوجد عدة طرق :*

- **الطريقة الأولي : حساب** $depositor \bowtie customer$ *أولاً ومن ثم استخدام النتيجة في حساب* $loan \bowtie (depositor \bowtie customer)$
- **الطريقة الثانية : حساب** $loan \bowtie depositor$ أولاُ، ومن ثم دمج النتيجة مع العلاقة *customer*
- **الطريقة الثالثة :** القيام بعمليتي الدمج سوية.
  - بفرض وجود فهرس على العلاقة loan بالواصفة *loan-number*، و *على customer في customer-name*
  - فمن أجل كل سطر من depositor نبحث عن الأسطر المرتبطة في customer والأسطر المرتبطة في loan
  - يجري بذلك اختبار كل سطر من depositor مرة واحدة فقط.