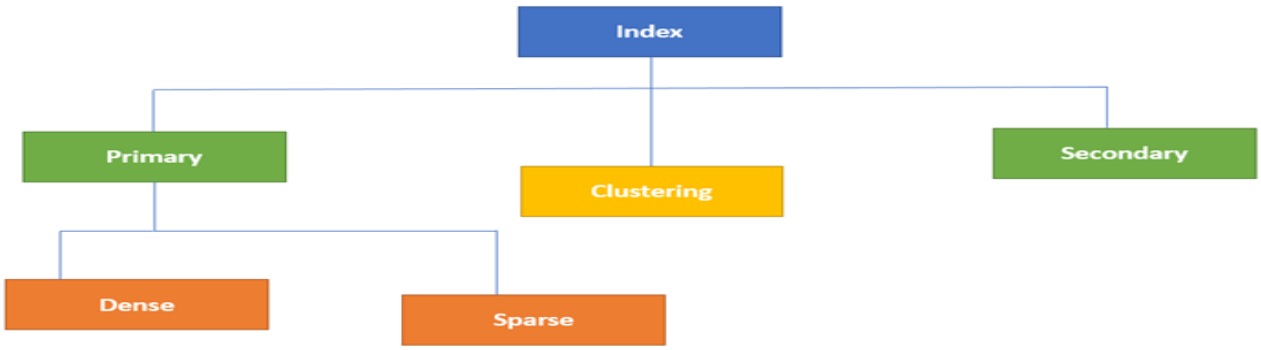
indexing in database and indexing type؟؟؟؟

اول حاجه هو يعني ايه Indexing

الـ **Indexing** هو تقنية بستخدمها عشان ارتب ال data في Database tables عشان البحث عن البيانات يكون أسرع. الـ Index بيكون جدول صغير فيه عمودين بيخزنوا نسخة من الـ primary key أو candidate key للجدول وعنوان مكان حفظ القيم بالجدول. الـ Index بيستخدم للبحث عن ال data بسرعة.

Types of Indexing in DBMS



عندنا نوعان من :Data indexes

1. (Primary Indexing)
2. (Secondary Indexing)

الـ **Primary Index** ده ملف مرتب بحجم ثابت في Database ، بيحتوي على معلومات عن الkey الرئيسي لل data وموقعها في الجدول. بيستخدم الـ Primary Index عشان اسهل search و update و deletion في الجدول بشكل سريع وفعال.

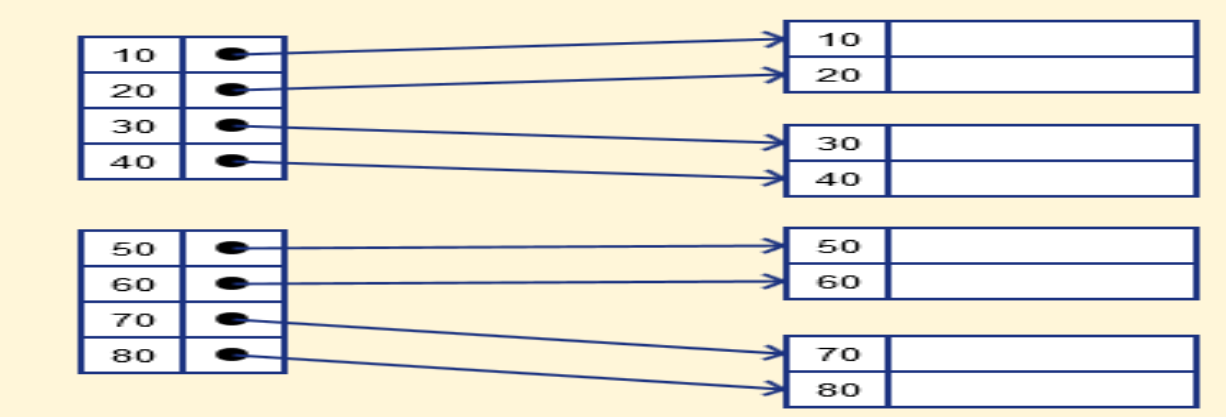
أما الـ **Secondary Index** فيقوم بإنشاء مؤشرات إضافية للبيانات الموجودة في الجدول، ويساعد على تسريع البحث والاستعلامات اللي لا تستخدم الkey الرئيسي كشرط بحث.

في الـ **Primary Index**، كل إدخالة في index table ترتبط بإدخالة واحدة فقط في الجدول، أما في الـ Secondary Index فيمكن أن ترتبط إدخالة في index table باكتر من إدخال في الجدول،عشان هيا بتستخدم للبحث بمعلومات غيرال key الرئيسي.

تنقسم Primary Indexing في Database إلى نوعين:

Dense Index و **Sparse Index**

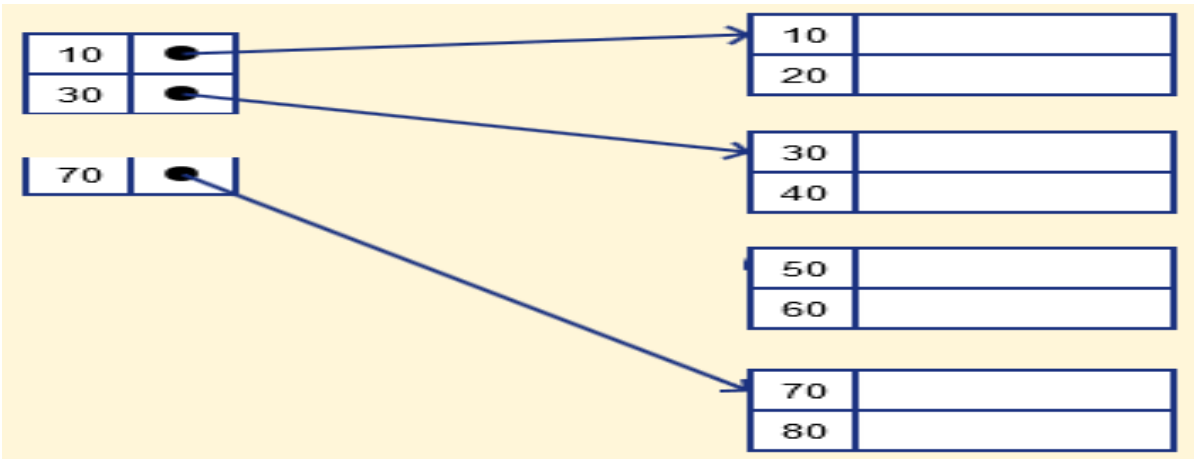
Dense Index هو الفهرس المركز في Database هو ملف خاص بيتم إنشاؤه لكل key بحث في Database ، حيث بيتم تخزين قيمة ال key وعنوان السجل المرتبط بيه في Hard Disk. يتيح النوع دا من ال indexes البحث بسرعة عن السجلات المطلوبة، ولكنه يستهلك مساحة أكبر لتخزين سجلات الفهرس على Disk.



ال **Sparse Index** في Database هو نوع من Data indexes اللي بيخزن Indexing records بس لبعض القيم في الملف. ده بيساعد على حل مشاكل ال Indexing الكتيره في DBMS، وبيخزن نطاق من أعمدة ال Index بنفس عنوان data block ، ولما يحتاج ياخد البيانات، بيجلب عنوان ال block.

بس ال **Sparse Index** بيخزن Indexing records بس لبعض قيم search key ، وده بيحتاج لمساحة أقل وكمان بيتطلب صيانة أقل لإدخال وحذف البيانات، بس بيبقى أبطأ من ال Dense Index في العثور على ال records .

مثال على Database index من نوع Sparse Index

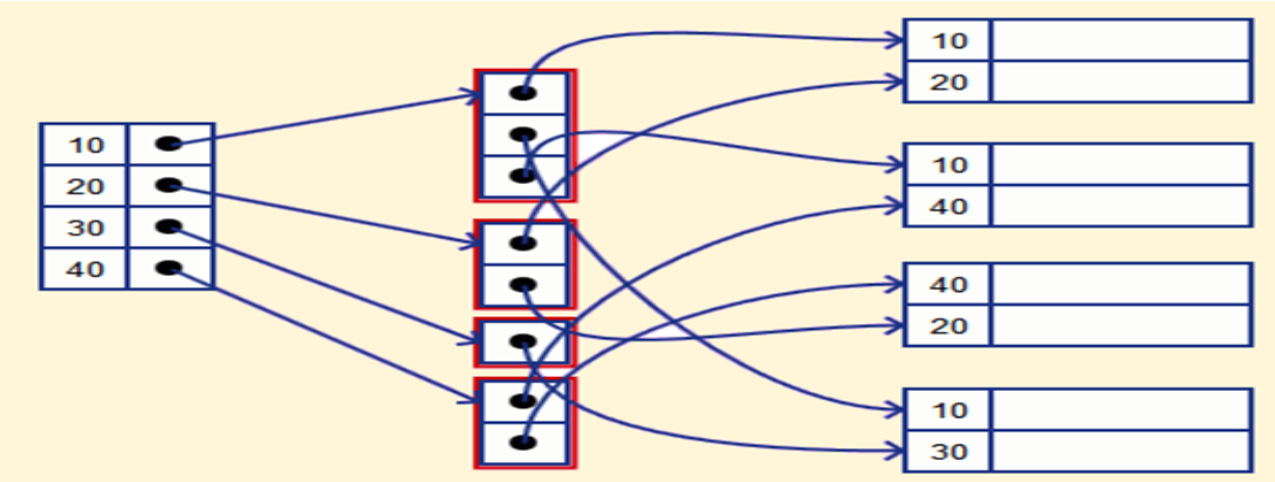


Secondary Index in DBMS

طيب يلا، الـ **Secondary Indexing** في Database هي تقنية بتساعد على البحث عن البيانات بشكل أسرع في الجداول الضخمة. وبتتم عن طريق اختيار field في الجدول اللي بيكون فيه كل قيمة فريدة لكل سجل ويجب أن يكون field دا Election key. وللتنفيذ بيتم انشاء Secondary Index وهو بيتميز باسم non-clustering index.

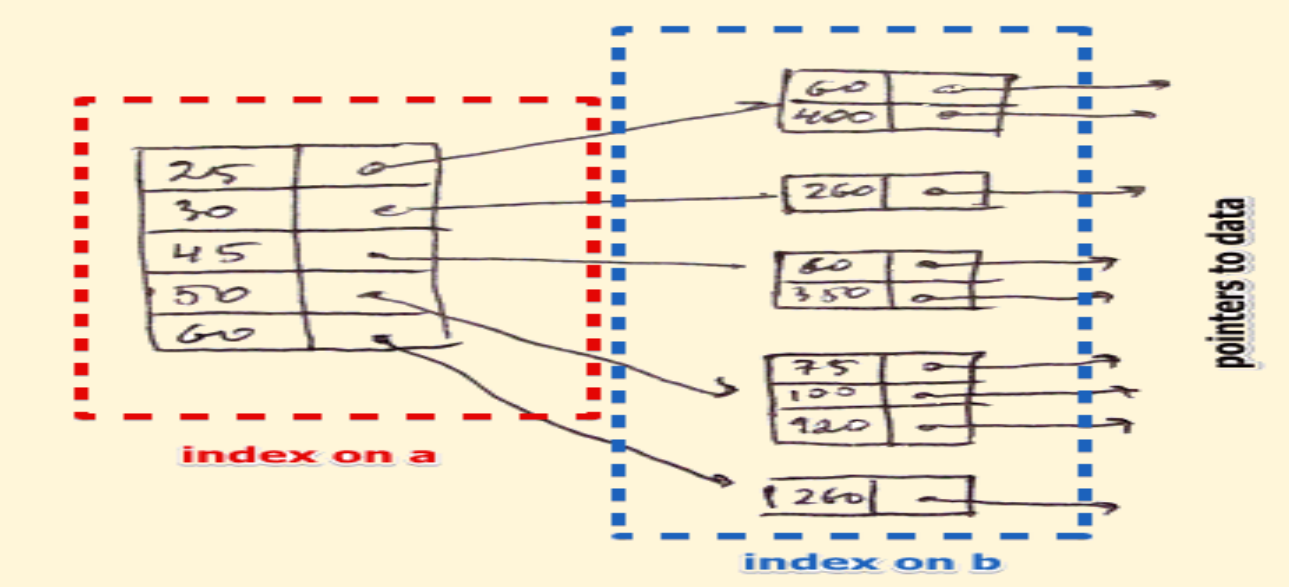
وبيستخدم هذه التقنية عشان تخفف من حجم التعيينات في الـ Indexing الخاصة بالجداول الكبيرة. وبيتم اختيار نطاق كبير من الأرقام بحيث يظل حجم التعيينات دايمًا صغير.

مثال علشان نفهم الفكرة، لو في Database لحسابات البنك والبيانات مرتبة بترتيب تصاعدي حسب رقم الحساب، وعايز تدور على جميع الحسابات اللي في فرع محدد من بنك ABC، يمكن إنشاء Secondary Index لكل بحث، واللي هو عبارة عن سجل بيشير لدلو يحتوي على روابط لكل السجلات



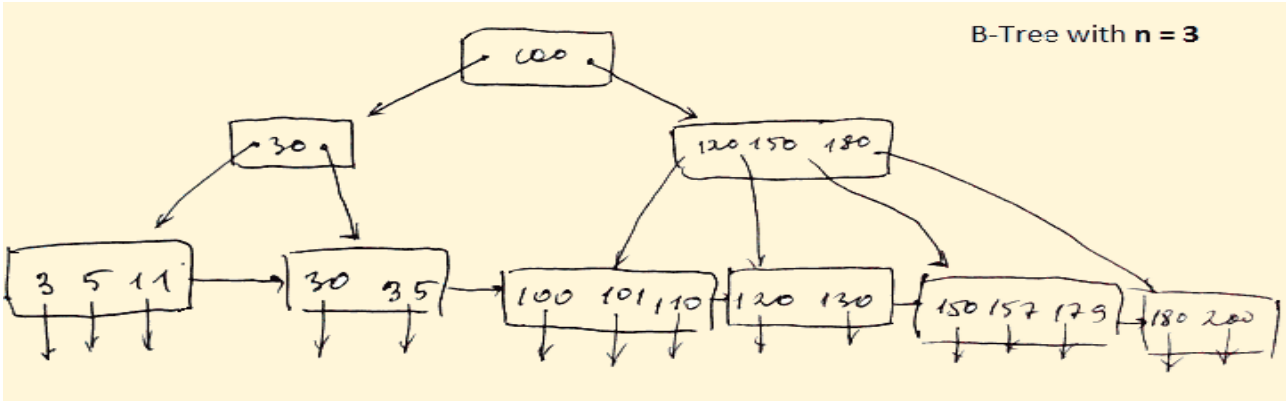
في ال Database الـ **clustered index** هو index بيتحط فيه البيانات نفسها بدل ما يتم استخدام index تشير لمواقع البيانات في جداول تانية. ال index ده بيتخلق أحيانا على أعمدة غير رئيسية غير فريدة لكل سجل، وهو بيساعد على الوصول للبيانات بشكل أسرع.

و **Multilevel Indexing** بيتخلق لو ال index الرئيسي مش بيتقدر يتخزن في الذاكرة. فبيتم تخزين ال index بشكل متدرج في عدة مستويات، وكل مستوى بيشير لمجموعة من السجلات. ده بيساعد على تقليل عدد عمليات الوصول للبيانات والحفاظ على البيانات مرتبة بشكل متسلسل وسريع الوصول.



الـ **Tree Index** هو نوع مستخدم كتير في الداتابيز، بيستخدم نظام Tree Index للبحث السريع عن البيانات. بيشتغل ال Tree Index عن طريق البحث الثنائي في ال Index وجميع أوراقه بتشير للمكان الفعلي للبيانات اللي هما بيشوفوها.

بالإضافة إلى ده، الـ Tree Index بيشبك جميع أوراقه مع بعض بقائمة روابط، اللي بيسمح ليه بدعم الوصول السريع للبيانات عن طريق رقم السطر، وكمان بدعم الوصول التسلسلي اللي بيمكن من الوصول للبيانات بترتيبها في الداتابيز.



الـ **Tree Index** دي بيتحكم في عدد القيم اللي في كل صفحة (ورقة) بين 2 و 4 قيم. وكل المسارات من الجذر لحد الورقة بتكون بنفس الطول تقريبًا. والأعقاب (Non-leaf nodes)، يعني العقد اللي مش جذر ولا ورقة، بيحتوي على بين 3 و 5 عقد أخرى. وكل عقدة مش بتكون لا جذر ولا ورقة، بتحتوي على بين n/2 و n عقدة. الـ B-Tree Index بتستخدم هذه الخواص لتنظيم البيانات بشكل فعال وسريع، بحيث تسهل عملية البحث عن البيانات وإدارتها في الـ DBMS.

الفرق بين charset و collation

الـ charset والـ collation دول مصطلحات مهمة في الداتابيز. الـ charset هو الحروف اللي ممكن تستخدمها في الداتابيز، والـ collation هو طريقة ترتيب الحروف والأرقام. يعني لو عندك داتابيز بتخزن فيها نصوص وأرقام، الـ charset هو الحروف اللي تقدر تستخدمها لتخزين البيانات دي، والـ collation هو الطريقة اللي تستخدمها لترتيب البيانات وتقارنها.

مثلاً، لو عندنا داتابيز بالعربي، هنستخدم charset اسمه UTF-8 عشان نقدر نخزن الحروف العربية والأرقام. وهنستخدم collation بالعربي عشان نقدر نرتب ونقارن الحروف والأرقام. وفي MySQL و PostgreSQL، الـ collation الشائعة هي utf8_general_ci، الـ utf8 هو الـ charset، والـ ci بتحدد الطريقة اللي هنستخدمها في ترتيب البيانات ومقارنتها.

وكمان، الـ charset والـ collation بيؤثروا على عملية البحث والفرز في الداتابيز، فلو اخترنا charset و collation غلط، ممكن يكون في مشاكل في البحث والفرز. فلو عايزين نستخدم الداتابيز بشكل صحيح، لازم نختار charset و collation مناسبين لنوع البيانات واللغة اللي بنستخدمها.

authentication and authorization

الـ**Authentication** هو عملية إثبات هويتك وأنت تقول الحقيقة عن نفسك. هذا يتم عن طريق التحقق من هوية الشخص أو الجهاز الذي تستخدمه. وفي بعض الأحيان يختصر بـ AuthN.

ويتم ذلك عن طريق إدخال اسم المستخدم وكلمة المرور، والتحقق منهما للتأكد من هويتك وأنك المستخدم المخول لك الدخول إلى النظام.

وتستخدم منصة Microsoft identity protocol OpenID Connect لمعالجة الـAuthentication وتوفير الأمان والحماية للمستخدمين عند تسجيل الدخول إلى المنصات المختلفة.

الـ **Authorization** هي عملية إعطاء الحق لشخص موثوق به بالقيام بشيء ما، مثل الوصول إلى معلومات معينة أو القيام بإجراء معين. وتحدد هذه الصلاحيات ما الذي يمكن للمستخدم الوصول إليه وما الذي يمكنه القيام به بالنسبة لهذه المعلومات. وغالبًا ما يتم تخفيضها لـ AuthZ.

يتم استخدام بروتوكول OAuth 2.0 في منصة هوية مايكروسوفت للتعامل مع عمليات الـ Authorization. وهو يعتمد على تفويض الوصول للمستخدمين وتخويلهم بإذن من صاحب المعلومات للوصول إليها. وعندما يقوم المستخدم بطلب الوصول إلى معلومات معينة، فإن الموقع الذي يحتوي على هذه المعلومات يطلب من المستخدم تفويض الوصول إلى هذه المعلومات، وعندما يتم التأكد من أن المستخدم موثوق به، يتم منحه الإذن المناسب للوصول إلى المعلومات.

بشكل عام، يساعد الـ Authentication والـ Authorization على ضمان أمان النظام ومنع الوصول غير المصرح به إلى المعلومات الحساسة.