جامعة بوليتكنك فلسطين
كلية تكنولوجيا المعلومات و هندسة الحاسوب

مشروع تخرج بعنوان

تجارب عملية لتطوير العمليتين التدريسية والتعلمية لمساقات تخصص تكنولوجيا المعلومات
(تطبيق على مساق تطبيقات الحوسبة السحابية)

إعداد:

هالة محمد حسن تلاحمة شيماء عبد الرحمن نتشه نور إسماعيل عمرو

بإشراف:

د.هاني صلاح أ.مهدي عطاونة

2018 / 2017

الملخص

هناك الكثير من المساقات التي درسناها في مسيرتنا التعليمية في تخصص تكنولوجيا المعلومات تحتوي على جوانب عملية, ولا يتم شرحها الا بالطريقة النظرية لعدة أسباب منها: عدم توفر المختبر الازم للمساق ليتمكن الطلاب من أداء الجانب العملي بأنفسهم. وأيضاً لا يوجد الوقت الكافي لمدرس المساق ليتمكن من أداء التجارب بوقت المحاضرة.

وهذا يؤدي إلى قلت استيعاب الطلاب لبعض المفاهيم أو المواضيع النظرية. ومن ضمن هذه المساقات مساق تطبيقات الحوسبة السحابية. التي تحتوي على العديد من التقنيات منها:

1. الزووكيبر (Zookeeper)
2. هادوب و ماب رديوس (Hadoop/MapReduce)
3. الجداول الموزعة (Distributed Hash Table (DHT))
4. كاسندرا (Casandra)

وبالاتفاق مع مدرس المساق اتضحَ أنها من أكثر التقنيات التي يواجه الطلاب صعوبة في استيعابها. وبالتالي وقع اختيار مشروعنا على هذه التقنيات لدعمها من أجل أن تصل فكرتها ومفهومها بشكل مثالي وصحيح .

ومن هنا كانت فكرة المشروع والدافع له, لتصميم موقع الكتروني يحتوي على التجارب أعلاه. الذي يوفر لكل تجربة: مفهومها الأساسي بطريقة سلسه. فيديو أنميشن يوضح طرية تسلسل التجربة. بالاضافه لفيديو يحتوي على تنفيذ والاعدادات الازمة للتجارب. وأيضاً ملفات تتضمن كل شيء لمساعدة الطالب للقيام بالتجارب دون مساعدة أحد .

ويجدر بالذكر هنا أن بعض التجارب تم إعطاءها لطلاب المساق في الفصل الماضي لأداءها وتجربتها بأنفسهم. ونجحت تقريباً بنسبة 90% وتم تعويض 10% المتبقية لأخذ الملاحظات الازمة للتحسين على التجارب.

المحتويات :

الأشكال

الجداول

الفصل الأول المقدمة

الفصل الأول المقدمة

1.1 الدافع

منذ بداية التعليم في جامعة بوليتكنك فلسطين، امتلئ هذا الصرح التعليمي بمجالات شتى يستقطب الطالب لارتيادها; لمرونتها وتماشيها مع ميول وتطلعات الشباب المزهرة.

حيث يوجد هناك أشكال عديدة من التكنولوجيا الحديثة التي تثير اهتمام الطلاب في جامعه بوليتكنك فلسطين والتي تكون مطلوبة في المساقات المطروحة في الجامعة, خاصة في كلية تكنولوجيا المعلومات وهندسة الحاسوب التي تهدف لبناء جيل ملم بهذه التكنولوجيا والتقنيات الحديثة من خلال طرحها في هذه المساقات.

ففي تخصص تكنولوجيا المعلومات في جامعة بوليتكنك فلسطين يوجد مساقات عديدة تتضمن جوانب علمية وعملية مهمة جدا من ضمنها : مساق تطبيقات الحوسبة السحابية, الذي يحتوي على كل ما به تجدد وتطور في عالم التقنية والتكنولوجيا التي تجذب الطلاب. ناهيك عن بعض عثراتها التي بجهودنا وتطلعاتنا حصدنا ثمرة معالجتها وتحسينها في مشروعنا هذا.

1.2 المشكلة

في هذا المشروع سوف يتم طرح مشكلة طريقة عرض وتدريس بعض المساقات في تخصص تكنولوجيا المعلومات ,حيث هناك العديد من المساقات في هذا التخصص تحتوي على جوانب عملية مهمة والتي يتم من خلالها فهم المحتوى النظري للمساق , ولكن بسبب كبر حجم المادة النظرية التي تأخذ النصيب الأكبر من وقت المحاضرات وعدم توفر المختبر اللازم للقيام بالتجارب العملية لا يكون هناك امكانية لعرض هذه التجارب بشكل كاف للطلاب بالتالي ينتج قلة استيعاب لدى الطالب لبعض المواضيع وعدم حصوله على الاستفاده الأكبر من المساق وهدفه التعليمي.

نتيجة لوجود هذه المشكلة سيتم التركيز في هذا المشروع على مساق الحوسبة السحابية, التي تحتوي على العديد من التقنيات, حيث سيتم إجراء مجموعة من التجارب على التقنيات الخاصة بها, والتي من شأنها أن توضح للطالب مبدأ عملها, وتوصيل الفكرة للطلاب بالشكل الصحيح . تم اختيار هذا المساق بسبب احتوائه على تقنيات مهمة تساعد في مجالات متعددة .

1.3 الحل

نتيجة للمشكله أعلاه , قمنا من خلال مشروعنا بتجارب عملية لتوضح التقنيات الموجودة في مساق الحوسبة السحابية حيث تم التركيز على أربع تقنيات من هذا المساق والتي سنتناولها في هذا المشروع وهي كالتالي:

1. الزوكيبر (Zookeeper)
2. هادوب و ماب رديوس (Hadoop/MapReduce)
3. الجداول الموزعة(Distributed Hash Table (DHT))
4. كاسندرا (Casandra)

فبإتمامها عمليا من قبلنا وفرنا موقع إلكتروني يحتوي شروحات نظرية مبسطة وفيديوهات تطبيقة توضح تجاربنا وتوصياتنا للسائل والمعني بالإستفادة، وبالأخص لطلاب المساق .

وتم اختيار التقنيات آنفة الذكر بالاتفاق مع مدرس المساق, التي اتضح أنها من أكثر تقنيات المساق التي يواجه الطلاب صعوبة بها وبالتالي قلت استيعابها.

1.4 الهدف

يعتبر هدف المشروع هدفاً تعليمياً بحيث يوفر لكل من الطالب والمدرس الطريقة المناسبة لعرض محتوى مساقات تكنولوجيا المعلومات والحصول على الفائدة من الهدف التعليمي لهذه المساقات. وبما انه يركز على مساق الحوسبة السحابية فإنه يهدف الى القيام بالعديد من التجارب العملية التي توضح التقنيات المذكوره أعلاه .

1.5 إسهامات المشروع

يسهم هذا المشروع في إثراء المساقات بالعديد من التجارب العملية التي لايكون هناك امكانية كافية لشرحها للطلاب من خلال العمل على تصميم وتنفيذ وتطبيق وتوثيق تجارب تغطي الجوانب العملية للمساق وتوضيح أهميتها بحيث سيتم تصميم هذه التجارب بشكل يسمح للطالب بتنفيذها(وحتى التعديل عليها)بشكل منفرد أو مع زملائه دون وجود حاجة كبيرة لتدخل المدرس.

وكذلك يسهم هذا المشروع بإضافة معرفة علمية واسعه لكل من الطلاب القائمين على المشروع وطلاب المساق من خلال التعرف على التقنيات بالشكل العملي وتجربتها مما يؤدي الى تقويتهم من الناحية العملية وبالتالي يكون لديهم فرصة أكبر في سوق العمل, حيث سيتم تصميم هذا المشروع لتطوير فهم المفاهيم الأساسية في مجال الحوسبه السحابية وتقنياته .

ويجدر بالذكر هنا أن بعض التجارب تم إعطاءها لطلاب المساق في الفصل الماضي لأداءها وتجربتها بأنفسهم. ونجحت تقريباً بنسبة 90% وتم تعويض 10% المتبقية لأخذ الملاحظات الازمة للتحسين على التجارب.

1.6 خلفية الحوسبة السحابية

الكثير منا سمع بالحوسبة السحابية تتردد كثيرا في الأوساط التقنية مؤخرا ولكن معناها يعتبر غامضا إلى حد كبير لدى الكثير منا. إذا بحثنا في معناها حرفيا فإن الحوسبة السحابية تعني أن الحاسبات تعمل في السحاب أو تبقى محلقه في الفضاء بينما يصل إليها المستخدمون.

لذلك دعونا في هذا القسم نتحدث أكثر عن الحوسبة السحابية, ونكشف بعض غموضها .

1.6.1 مفهوم الحوسبة السحابية

مع التطور الحادث فى التقنيات المتاحة من خلال شبكة الانترنت، وتسارع تدفق البيانات عبر الانترنت، عملت العديد من المؤسسات والشركات العملاقه على إتاحة تطبيقاتها عبر الانترنت باستخدام الحوسبة السحابية، هذه التقنية أفادت المستخدمين على نطاق واسع بتوفير النفقات . وتعد الحوسبة السحابية بمثابة تكنولوجيا تعتمد على نقل المعالجة ومساحة التخزين الخاصة بالحاسوب إلى ما يسمى السحابة، وهي عبارة عن أجهزة خوادم يتم الوصول إليها عن طريق الانترنت، لتتحول البرامج من منتجات إلى خدمات، ويتاح للمستخدمين الوصول إليها عبر الانترنت.

وجميع محتويات الحوسبة السحابية تكون داخل مزارع ضخمة تسمى بمراكز البيانات (dara centers), وهي عبارة عن أجهزة وبرامج يستخدمها المزودون من أجل تقديم خدمات الحوسبة إلى العملاء, ليتمكنوا من الوصول والاستفادة منها.

1.6.2 أنواع الحوسبة السحابية

يوجد أربعة أنواع أساسية من الحوسبة السحابية يمكن تلخيصها بالآتي :

1. سحابة خاصة (Private Cloud) : هذا النوع من السحابات يكون عادة داخل المنشأة بحيث يمكن الوصول إليها من خلال الشبكة المحلية أو من خلال الإنترنت ويتم تقديم الخدمات للمستفيدين بشكل تلقائي كما يمكن أن تكون موجودة لدى شركة إستضافة, و في جميع هذه الحالات تستطيع المنشأة مراقبة مكونات البنية التحتية و التحكم فيها .

2. سحابة عامة (Public Cloud) : وهي عبارة عن خدمات تجارية يقدمها مزود الخدمة لعملاء متعددين و تكون موجودة في مكان بعيد عن العميل و هي وسيلة لتوفير التكاليف و ربح الوقت والجهد.
3. سحابة مختلطة / هجينة (Hybrid Cloud) : وهي تجمع بين خصائص السحابة الخاصة و العامة. إذ يمكن لمنشأة أن يكون لها سحابة خاصة تقوم من خلالها بتوفير بعض الخدمات للمستفدين, بينما تلجأ إلى حلول السحابة العامة لتأمين خدمات أخرى .

1.6.3 خدمات الحوسبة السحابية

تتوفر الحوسبة السحابية بشكل أساسي إلى ثلاث خدمات, ومنها :

1. التطبيقات كخدمة (Software as a Service (SaaS)) : تقدم البرمجيات كخدمة. يمكن أن نذكر كمثال ما تقدمه شركة جوجل من خلال حزمة "تطبيقات جوجل التي تشمل برنامج تحرير النصوص و التقويم . كما يمكن إدراج مفهوم سطح المكتب كخدمة ضمن هذا نموذج حيث تكون التطبيقات متاحة للمستخدم و لكن أيضا الجهاز نفسه يكون متاحا كخدمة بما في ذلك نظام التشغيل و سطح المكتب و يمكن الوصول إليه من أي مكان.

2. المنصة كخدمة (Platform as a Service (PaaS)) : تقدم منصة الحوسبة كخدمة و تكون أداة البرمجة نفسها مستضافة على السحابة و يمكن الوصول إليها من خلال المتصفح. يتيح هذا النوع من الخدمات للمبرمجين بشكل عام إمكانية تطوير و بناء تطبيقات ويب دون الحاجة إلى تثبيت أي برامج أو أدوات على أجهزتهم.

3. البنية التحتية كخدمة (Infrastructure as a Service (IaaS)) : تقدم البنية التحية كخدمة. وتمكن المؤسسات المتوسطة و الصغيرة من إدارة البيئة التقنية التحتية و البرامج عن طريق الإنترنت بطريقة سهلة و آمنة دون الحاجة إلى أن تكون لديهم مراكز بيانات مكلفة .

1.6.4 سبب نجاح الحوسبة السحابية

يوجد دوافع كثيرة تحث المستخدمين على استخدام الحوسبة السحابية وبالتالي نجاحها ومنها :

- توفير الوقت و الجهد داخل بيئة العمل.
- خفض التكاليف بشكل عام(رخص البرمجيات , التخزين , الخوادم) .
- المرونة في سعة التخزين و الموارد بشكل عام .
- توفير التحديثات للبرامج و التطبيقات .

1.6.5 تحديات الحوسبة السحابية

بالرغم من النجاح والفائدة الكبيرة للحوسبة السحابية, إلا أنه يوجد بعض العقبات عند استخدامها, ومنها :

- تتطلب إتصالا مستمراً بشبكة الانترنت .
- المخاطر المتعلقة بخصوصية وأمن البيانات .
- ضمان مستوى الحماية .

1.7 بقية الفصول

الفصل الثاني الزوكيبر (Zookeeper) : في هذا الفصل سيتم التحدث عن موضوع الزووكيبر ومفهومه وهيكليته ووومكوناته ، وايضا التجربة الخاصة به وكيفية تطبيقه.

الفصل الثالث هادوب (Hadoop) ماب رديوس(MapReduce) : وفي هذا الفصل سيتم التحدث عن الهادوب الذي يتم من خلاله تنفيذ فكرة الماب رديوس وتوضيحة من خلال مثال عداد الكلمات .

الفصل الرابع الجداول الموزعة ((Distributed Hash Table (DHT) : في هذا الفصل سيتم التحدث عن موضوع الجداول الموزعة ومفهومها وخائصها, وأيضا التجربة الخاصة بها وكيفية تطبيقها .

الفصل الخامس كاسندرا (cassandra):في هذا الفصل سوف يتم التحدث عن إحدى طرق التخزين على السحابة وتوضيح مفهومها وخصائصها وأيضا توضيح كيفية عملها والتجربة الخاصة بها.

الفصل الثاني : زووكيبر (Zookeeper)

الفصل الثاني : زووكيبر (Zookeeper)

في هذا الفصل سيتم التحدث عن موضوع الزووكيبر ومفهومه وهيكليته ووومكوناته ، وايضا التجربة الخاصة به وكيفية تطبيقه .

2.1 مفهوم الزووكيبر

الزووكيبر هي عبارة عن خدمة تستخدم من قبل مجموعة من السيرفرات ، للتنسيق فيما بينها والحفاظ على عملية تقاسم البيانات وتوزيعها على هذه الاجهزة ، حيث ان عملية الادارة والتنسيق هي عملية معقدة ، ومن هنا أتى مفهوم الزوكيبر ليحل هذه المشكلة مع بنية بسيطة ، حيث يسمح للمطورين التركيز على منطق التطبيق الأساسي دون أن يشعره بالقلق حول الطبيعة الموزعة لهذا التطبيق ، فقد أصبح الوصول الى تطبيقات الزوكيبر بطريقة سهلة وقوية ، وقد أصبح الزوكيبر معياراً للخدمة المستخدمة من قبل اش باس (hbase) والهادوب (hadoop) والأطر الموزعة الاخرى ، على سبيل المثال : يستخدم اش باس الزوكيبر لتتبع حالة البيانات الموزعة.

قبل المضي قدما من المهم أن نعرف شيئا حول التطبيقات الموزعة ، حيث يمكن تشغيل التطبيقات الموزعة على أنظمة متعددة في شبكة في وقت معين (في وقت واحد) من خلال التنسيق فيما بينهم لإكمال مهمة معينة بطريقة فعالة وسريعة ، حيث أنه في التطبيقات الغير الموزعة تستغرق هذه المهام المعقدة وقتا طويلا (ساعات) حيث أن التشغيل يتم في نظام واحد والذي يمكن ان يتم خلال دقائق في التطبيقات الموزعة باستخدام قدرات الحوسبة السحابية .

2.2 أبرز مصطلحات الزووكيبر

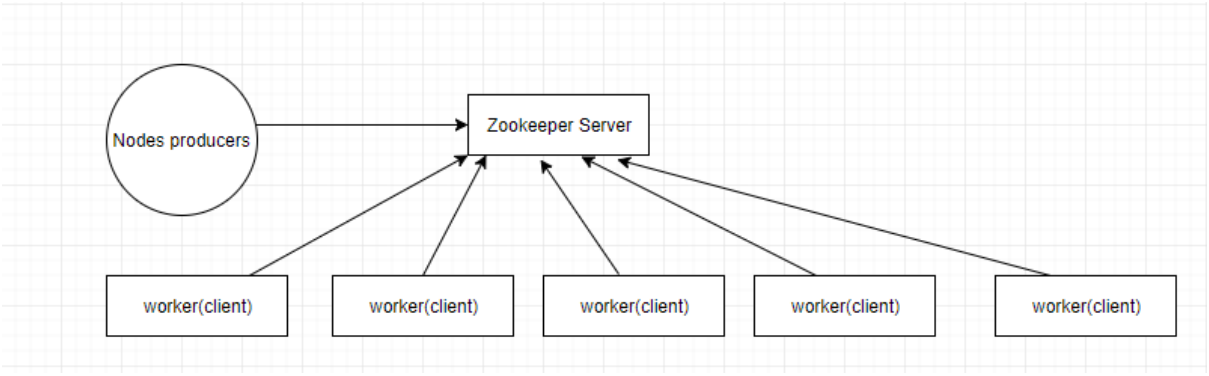
يوجد مجموعة من المصطلحات المهمة لفهم وتوضيح خدمة الزوكيبر، يجب الاطلاع عليها في البداية لتوضيح الأمور اللاحقة بشكل أفضل ومنها :

- العميل (client) : يقوم باستخدام خدمة الزووكيبر .
- الخادم (server) : يقوم بتوفير خدمة الزووكيبر .
- (Znode) : عقدة البيانات في ذاكرة الزوكيبر، نظمت على شكل هرمي تسمى شجرة البيانات .
- تحديث / كتابة (update / write) : أي عملية تعديل لشجرة البيانات .
- (session) : يقوم العميل بإنشاء جلسة اتصال للزوكيبر .

2.3 هيكلية الزووكيبر

الرسم البياني الموضح بالشكل (1) يصور هيكلية الخادم والعميل (Client-Server Architecture) للزوكيبر .

كما هو موضح العميل يقوم بالاتصال مع أي خادم متوفر. والخادم القائد يقوم بالادارة والتنسيق والاتصال فيما بينهم .



شكل 1:هيكلية الزوكيبر

الجدول رقم (1) يوضح هيكلية الزوكيبر، ويحتوي على الأجزاء التالية : العميل ، الخادم .

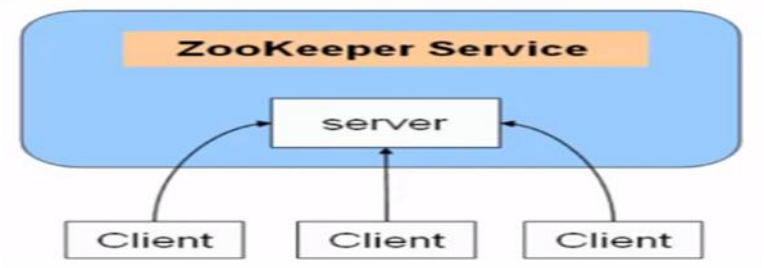
الجزء	الوصف
العميل	هو واحد من العقد يقوم بالوصول إلى المعلومات الموجودة بالخادم . خلال فترة زمنية محددة ، كل عميل يقوم بإرسال رسالة إلى الخادم لكي يعرف أن العميل على قيد الحياة ، وبالمقابل يرسل الخادم تأكيد لاتصال العميل به .
الخادم	واحدة من العقد في مجموعة زووكير، ويوفر جميع الخدمات للعملاء ، يعطي إقرار للعميل بأن الخادم على قيد الحياة.

جدول 1:هيكلية الزوكيبر

2.4 توزيعات الزووكيبر

2.4.1 التوزيعة المستقلة (Standalone mode)

هو عبارة عن جهاز مستقل قادر على العمل بشكل مستقل عن الأجهزة الأخرى ، وهذا يعني أنه غير مدمج مع أجهزة أخرى ، أي انه يحتوي على خادم زووكيبر واحد (server) يعمل على توزيع البيانات على مجموعة من الاجهزة (clients) كما هو موضح في الشكل رقم (2) :

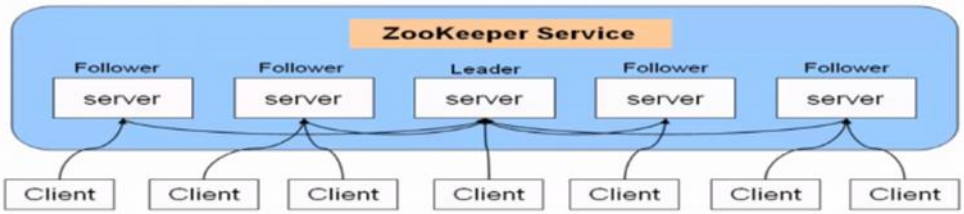


شكل 2:التوزيعة المستقلة.[1]

لكن يحتوي على مشكلة إذا تعرض الخادم الوحيد إلى خلل أو عطل فإن جميع الأجهزة التابعة له سوف تتوقف عن العمل ، لعدم وجود خادم بديل.

2.4.2 توزيعة النسخ المتماثلة (Replication mode)

هي عبارة عن مجموعة من الأجهزة (servers) التي تعمل مع بعضها البعض بحيث يكون هناك جهاز مسؤول عن باقي الاجهزة التي يعمل معها ويعمل على ادارة العملية فيما بينهم وتوزيع البيانات بالتناوب فيما بينهم ، حيث انه يتم توزيع البيانات على مجموعة من الاجهزة (client) كما هو موضح في الشكل رقم (3) :

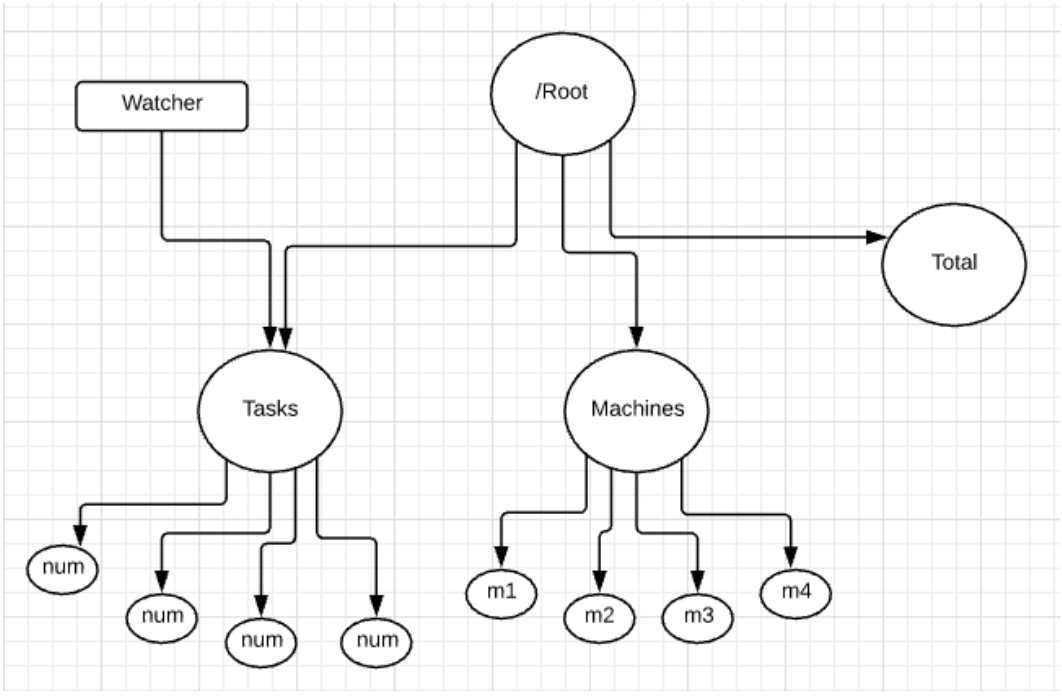


شكل 3:توزيعة النسخ المتماثل[1]

تتميز هذه التوزيعه بالمرونة العالية (فشل سيرفر واحد لا يؤثر على أداء النظام) حيث أنه الجهاز الواحد (client) يقوم الاتصال بخادم واحد فقط وفي حالة فشل الخادم يقوم بالاتصال بشكل تلقائي بخادم آخر .

Znode 2.5

يتم تنسيق العمليات الموزعة من خلال الزينود (znode) المشتركة المرتبة بشكل هرمي ، ونظمت بشكل يشبه ملفات نظام يونكس . وتتكون الزينود من سجلات البيانات ويمكن التلاعب بها من قبل العملاء (client) من خلال API ، ويمكن أن يكون لها تفرعات تسمى children.



شكل 4:التسلسل الهرمي للزوكيبر (Znode)

2.6 المراقب (watch flag)

عندما يقوم العملاء باصدار عمليات القراءة أو التحديث على الزينود مع وجود المراقب ، فإنه يقوم بإبلاغ كل من الخادم (server) والعميل (client) عند تغيير المعلومات على الزينود ، وليس عند إضافة بيانات جديدة .

ويتم تفعيله لمره واحده تكون مرتبطة بجلسة معينة .

2.7 الجلسة (Session)

تبدأ الجلسة عندما يقوم العميل بالاتصال مع خادم الزوكيبر ، وبالتالي تحتوي الجلسة على فتره زمنية تكون مرتبطة بها ، يعتبر زوكيبر أن العميل معيب أو فاشل إذا لم يتلق أي شيء من جلسته لأكثر من تلك الفترة الزمنية ، وتنتهي الجلسة إما عندما يقوم العميل بالاغلاق الصريح للجلسة أو من خلال كشف الزوكيبر عند وجود خلل أو عيب في العميل ، في غضون جلسة ، يلاحظ العميل سلسلة من تغييرات التي تعكس تنفيذ عملياته ، تمكن الجلسات العميل من الانتقال بشفافية من خادم إلى آخرضمن مجموعة زوكيبر، وبالتالي تستمر عبرخوادم زوكيبر.

Client API 2.8

نقدم أدناه العديد من الاوامر ذات علاقة بالزوكيبر API ، و دلالات كل أمر.

- Creat (path, data, flags)
إنشاء زينود مع اسم المسار، وتخزين البيانات ، وإرجاع اسم الزينود الجديد . flags تقوم بتحديد نوع الزينود.
- Delete (path, version)
يحذف الزينود إذا كان الإصدار مساوياً للإصدار الفعلي من زينود .
- Exists (path, watch)
في حال وجود الزينود يعرض قيمة صحيحة (true) ، وقيمة خاطئة (false) في حالات مخالفة .
- getData (path, watch)
لعرض البيانات المخزنة في الزينود ، لا يتم ضبط المراقب (watch) ما لم يكن هناك زينود .
- setData (path, data, version)
إعادة كتابة بيانات الزينود، إذا كان الإصدار مساوياً للإصدار الفعلي من زينود .
- getChildren (znode, watch)
إرجاع مجموعة فرعية من الزينود .

2.9 فوائد الزووكيبر

1. تبسيط عملية التنسيق الموزعة .
2. توثيق البيانات .
3. التسلسل في العمليات ، والتأكد من تشغيل الخدمة باستمرار .

2.10 الادوات المستخدمة

هناك عدة ادوات تم استخدامها لمساعدتنا على انجاز المشروع بسهولة ومنها :

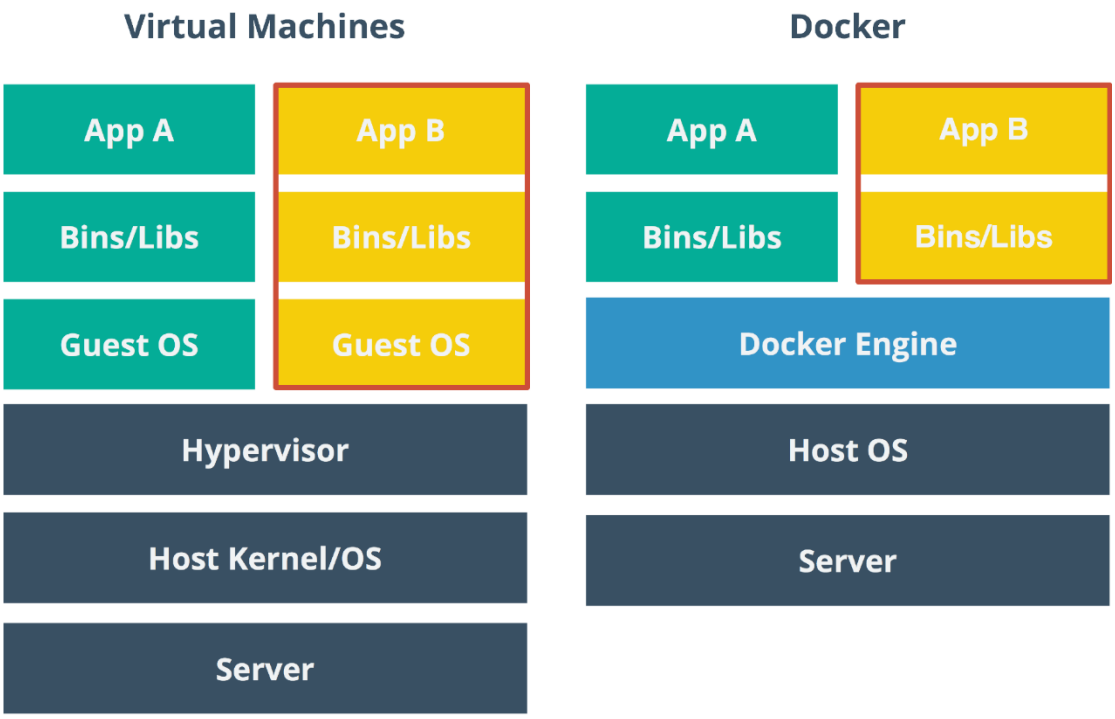
2.10.1 دوكر(Docker)

هذة التكنولوجيا تسمح بتشتغيل عدة تطبيقات مختلفة على نفس نظام التشغيل ويكون كل تطبيق منفصل تماما عن التطبيق الاخر ، وكل تطبيق يكون معه كل ما يحتاجة لكى يعمل دون أن يؤثرعلى تطبيق اخر ، كل هذا يحصل عن طريق ان يكون كل تطبيق داخل حاويات منعزلة (isolated Container) .

يمكن أن يتسائل القارئ أنه يمكننا عمل ذلك من خلال أنظمة التشغيل المختلفة مثل الويندوز او الليونكس ويمكن تشغيل اكتر من تطبيق على نفس نظام التشغيل ، لكن الفرق في هذه الحالة تكون ملفاتهم ومكوناتهم متداخلة مع نظام التشغيل ولا يمكن فصل هذة التطبيقات عن بعضها ولا نقلها لانها اصبحت جزءاً متداخلاً متداخل مع نظام التشغيل وتطبيقات اخرى .
اما فى حالة الحاويات (Container) كل تطبيق منفصل تماما فى ملفاتة واسمه .

تم توظيف الدوكر من خلال المشروع عن طريق تنصيب تطبيقات وكل ما تضمنه من ملفات خاصة تحتاجه مثل تنصيب الزووكيبر ، وبالتالي حل مشكلة توحيد إصدارات البرنامج ، بتوفره كحزمة كاملة .

فى الشكل رقم (5) سوف نجد بأن المحاكاه الافتراضية تقوم بعمل نظام تشغيل كامل لكل تطبيق ، وبالتالي يكون كل تطبيق منفصل عن الثاني ، وجميعهم يعملون في النهاية داخل مراقب أجهزة افتراضية (Hypervisor) .أما في حالة الحاويات نحتاج للنظام تشغيل واحد يتعامل مباشرة مع (hardware) ثم يقوم ببناء التطبيقات الخاصة بنا داخل الحاوية على نفس نظام التشغيل .



شكل 5:الفرق بين المحاكاة الافتراضية (virtualization) والدوكر (Docker)[2]

Node.js 2.10.2

اعتمدنا في كتابة الاكواد البرمجية على لغة node.js التي تنتمي للغة البرمجة الجافا سكريبت (javascript) ، حيث أعطت للجافا سكريبت بعدا آخر فأصبح بإمكاننا استعمال الجافا سكريبت خارج المتصفحات والتعامل مع الخوادم مباشرة (كما فعلنا ضمن مشروعنا) وبالتالي تطوير تطبيقات سريعة وفعالة باستعمال لغة واحدة فقط وهي الجافاسكريبت.

وأيضا عند استخدام node.js احتجنا إلى تنزيل مدير الحزم (package manager) المعروف عنه NPM.

وتحتوي الحزمة في node.js على كافة الملفات التي تحتاجها الوحدة نمطية (module) وهي عباره عن مكتبات جافا سكريبت .

2.11 تجربة الزووكيبر

2.11.1 وصف التجربة

صممت هذه التجربة للقيام بعملية جمع بسيطة وذلك باستخدام خادم الزوكيبر (Zookeeper Server) حيث سيتم من خلال هذه التجربة القيام بجمع ارقام مختلفة مخزنة على (nodes) موجودة على خادم الزوكيبر وتخزين النتيجة لهذه العملية على (node) خاصة بالمجموع الكلي موجوده على خادم الزوكيبر.

بشكل مفصل : تجربة الجمع هذه قائمة على وجود مجموعة هائلة من الأرقام المخزنة داخل نقاط (nodes) موجودة على خادم الزوكبير حيث أن كل نود تحتوي على رقم واحد وسوف يتم التعامل معها وكأنها مهمة مستقلة بحيث يتم تنفيذها بعد ذلك يتم التخلص منها حتى لا يتم تنفيذها مرة أخرى من قبل مستخدم آخر. هذه الأرقام سوف يتم جمعها من قبل عدد كبير من المستخدمين بحيث أن كل مستخدم سوف يقوم بقراءة رقم واحد من النقاط الموجودة على الخادم وإضافتها الى المجموع وبعد الانتهاء سيقوم بتخزين هذا المجموع على نود خاصة بالمجموع الكلي على خادم الزوكيبر ، وسيقوم بطلب رقم جديد.

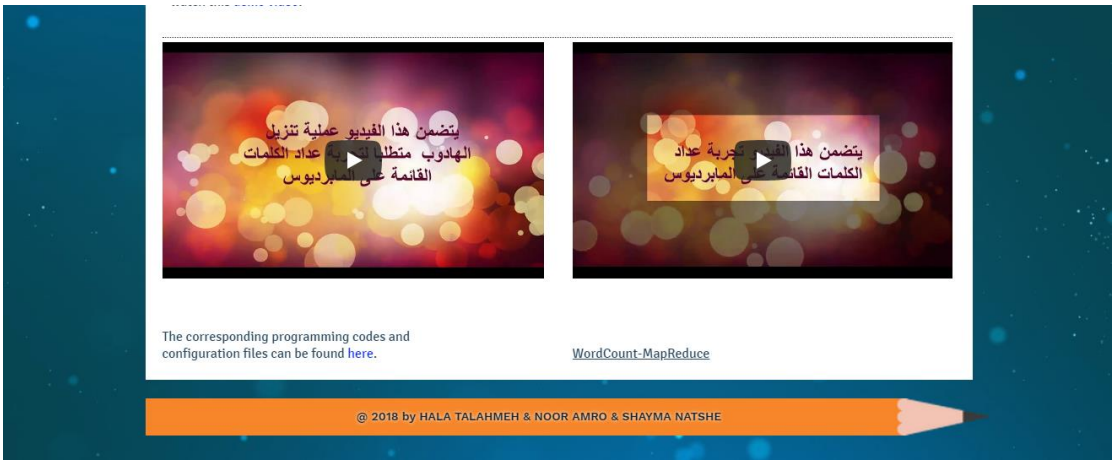
سيكون هناك ملفان أساسيان:

- الملف الأول : workers.js

عند إتصاله مع الخادم سوف يقوم بإضافة نفسه إلى النود التي تقوم بعملية الجمع والتي تدعى (machine nodes) حيث أنها توضح النود التي تكون فعالة وتقوم بعملية الجمع ، بعد ذلك يقوم بفحص ما اذا كانت النود التي تحتوي على هذه الارقام والتي تدعى (Task node) موجودة أم لا حيث اذا لم تكن موجودة سوف يتوقف ولن يقوم بأي عمليه ، في حال ما اذا كانت موجودة سوف يقوم بإحضار النود التي تحتوي على الارقام وتدعى (child node) اولا ثم سيقوم بإحضار المسار (path) الخاص بال child الاول وتمريره للاقتران (getvalue) حتى يرجع القيمه المخزنة في هذا المسار ، بعد احضار القيمة سيتم تمريرها الى اقتران (process task) حيث تم اضافتها الى المجموع ، بعد ذلك تم تمرير المجموع الى الاقتران (updatetotal) يعمل على فحص ما اذا كانت نود موجودة ام لا اذا لم تكن موجودة يقوم بإنشائها في حال كانت موجودة يحضر القيمه الموجودة في هذه نود وإضافة المجموع اليها ، بعد ذلك سيتم تخزين هذا المجموع على (total node) الخاصة بالمجموع الكلي .

- الملف الثاني : Values.js

هو الملف الذي يقوم بإنشاء (child nodes) التي تحتوي على الارقام .



شكل 6:مبدأ عمل التجربة من خلال flowchart

2.11.2 الهدف من التجربة

إن الهدف الأساسي من هذه التجربة هو هدف تعليمي ، حيث تهدف الى تسهيل العملية التدريسية على المدرس بحيث يمكنه تقديم المعلومات له بشكل سلس يسهل عملية الاستيعاب لدى الطالب دون أن يكون لديه أي شكوك في عدم قدرته على الاستيعاب أو عدم تمكنه من فهم المعلومات التي يحاول المعلم ايصالها له ، في هذه التجربة وهي تجربة تقنية الزووكيبر ستساعد الطالب على استيعاب مبدأ عمل تقنية الزووكيبر والخصائص التي يوفرها بسهولة .

الفروقات الناتجة من عملية اجراء التجربة باستخدام خادم الزوكيبر واستخدام الخادم العادي :

عملية الجمع التي نعمل عليها لو تم تنفيذها في الخادم العادي فإن العملية تكون صعبة ومعقدة بحيث يتم كتابة أكواد كثيرة مما يصعب على المبرمج التعامل معها وفي حال ما اذا حدث عطل في السيرفر سيؤدي الى انقطاع هذا الخادم مما يحتاج الى متابعة مباشرة من قبل التقنيين ليتم اعادته الى العمل مرة اخرى وهذه العملية ليست ببسيطة .

أما لو تم استخدام عملية الجمع باستخدام خادم الزوكيبر فان العملية تكون بسيطة مما يسهل التعامل معها من قبل المبرمج بحيث لا تتطلب كتابة اكواد كثيرة ومعقدة انما كتابة اكواد بسيطة وسهلة الفهم على كل من المبرمج والمستخدم ، حيث ان خادم الزوكيبر يقوم بالجزء الاكبر من العمليات ، في حال اذا حدث خلل في السيرفر سيتم نقل قاعدة البيانات الخاصه بهذا الخادم الى خادم اخر في مكان من العالم وذلك من خلال مركز البيانات (data center) ، مما يمكن المستخدم من التحكم في زيادة او نقصان في اعدادات السيرفر على حسب الحاجة وبكل سهولة .

2.11.3 المتطلبات اللازمة لتطبيق التجربة

تتمثل متطلبات التجربة كالتالي:

- برنامج لإستعراض الأكواد الخاصة بالتجربة مثل Visual Studio Code أو Sublime
- تم تطبيق هذه التجربة على نظام تشغيل لينوكس (Linux Ubuntu).

2.11.4 تنفيذ التجربة

حتى يتمكن الطالب من تنفيذ هذه التجربة يوجد أمران مهمان يتم من خلالهما تشغيل التجربة ، الأمر الأول هو node Values.js والذي يتم من خلاله انشاء النود التي تحتوي على الارقام ، الأمر الثاني هو node workers2.js الذي يقوم بعملية الجمع .

2.11.5 أوامر التجربة

فيما يلي ذِكْر للأوامر التي سيتم تنفيذها قبل القيام بتنفيذ التجربة

قم بتحديث حزمة apt	$ sudo apt-get update
تثبيت الحزم للسماح ل apt باستخدام مستودع عبر HTTPS	$ sudo apt-get install \ apt-transport-https \ ca-certificates \

curl \ software-properties-common	
$ curl -fsSL https://download.docker.com /linux/ubuntu/gpg | sudo apt-key add -	إضافة مفتاح GPG الرسمي ل Docker
$ sudo add-apt-repository \ "deb [arch=amd64] https://download.docker.com /linux/Ubuntu\ $(lsb_release -cs) \ stable"	استخدم الأمر التالي لإعداد المستودع المستقر(stable repository)
$ sudo apt-get update	قم بتحديث حزمة apt
$ sudo apt-get install docker-ce	قم بتثبيت أحدث إصدار من Docker CE
$ sudo docker run hello-world	تحقق من تثبيت Docker CE بشكل صحيح عن طريق تشغيل صورة hello-world
$ sudo docker pull zookeeper	تحميل صورة zookeeper على docker
$ sudo apt-get install node.js	تحميل node js على docker
$ npm install zookeeper –save	تحميل npm على docker
$ npm install zookeeper –save	تحميل الزووكيبر على docker
$ sudo docker run --name zk-server -p 2181:2181 -p 2888:2888 -p 3888:3888 --restart always -d zookeeper	حتى يتم تشغيل ملفات الزووكيبر

جدول 2:أوامر تجربة الزووكيبر

2.11.6 نتائج تنفيذ التجربة

الشكل (7) يوضح عملية إنشاء child node داخل Task node وهذه العملية تتم في ملف Values.js .

```
hala@hala-HP-ProBook-4530s:~/demo$ node Values.js
connected
created zk node /Tasks/0000000010
created zk node /Tasks/0000000011
created zk node /Tasks/0000000012
created zk node /Tasks/0000000013
created zk node /Tasks/0000000014
```

شكل 7:إنشاء child

الشكل (8)،(9) يوضح عملية إحضار child node من Task node وجلب القيمة الموجودة في child node حتى يتم جمعها ووضعها في Total node الا وهو node الخاص بالمجموع الكلي ,حيث أن هذه العملية تتم في ملف workers.js .

```
hala@hala-HP-ProBook-4530s:~/demo$ node workers2.js
client_id null 1000000c9540001
tasks node exist
Childrens [ '0000000012',
  '0000000011',
  '0000000014',
  '0000000013',
  '0000000010' ]
first child: /Tasks/0000000012
get value function: /Tasks/0000000012
first child: /Tasks/0000000011
get value function: /Tasks/0000000011
first child: /Tasks/0000000014
get value function: /Tasks/0000000014
first child: /Tasks/0000000013
get value function: /Tasks/0000000013
first child: /Tasks/0000000010
get value function: /Tasks/0000000010
updateTotal 3
updateTotal 2
updateTotal 5
updateTotal 4
updateTotal 1
Get total:  data 2005
total= 3008
```

شكل 8:عملية التجمع

```
updateTotal 1
Get total:  data 2005
total= 3008
version 2
Get total:  data 2005
total= 3007
version 2
Get total:  data 2005
total= 3010
version 2
Get total:  data 2005
total= 3009
version 2
Get total:  data 2005
total= 3006
version 2
type 4 state 3 path /Tasks
```

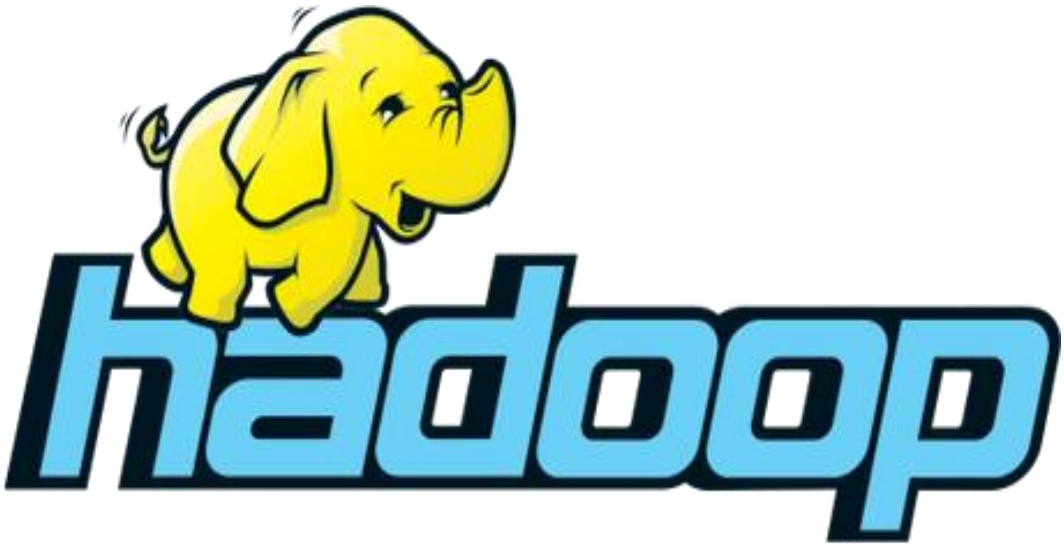
شكل 9:عملية الجمع

2.11.7 ملحقات التجربة

لقد تم تزويد هذه التجربة بالعديد من الملحقات التي تعتبر مساعدة للطالب حتى يتمكن من فهم هذه التجربة ، وتمثلت هذه الملحقات بالآتي :

- فيديو توضيحي باستخدام الانيميشن (Animation Video)
  8ND5s9v0Aj9https://www.youtube.com/watch?v=
- فيديو خاص باعدادات وتنفيذ التجربة (Configration & Execution video)
  s302xgk&t=3https://www.youtube.com/watch?v=yoI-ELt

الفصل الثالث : هادوب (Hadoop) ماب رديوس(MapReduce)



الفصل الثالث : هادوب (Hadoop) ماب رديوس (MapReduce)

تتدفق البيانات بأشكال متنوعه وصور مختلفه من خلال الشركات الكبرى أو عبر الشبكة العنكبوتية . وهذا الكم الهائل من البيانات تحت ظروف وأغراض معينة يحتاج الى معالجة . على سبيل المثال ما تقوم به شركة الفيسبوك من معالجة بيانات المليارات من المستخدمين .

إن هذه البيانات اذا تمت معالجتها على جهاز واحد فسوف تأخذ شهوراً للمعالجة بالإضافة الى مساحات تخزين كبيرة ; لذلك يتم اللجوء الى استخدام مجموعة من أجهزة الحواسيب التي تتصل مع بعضها البعض من خلال شبكة موصلة فيما بينها والتي قد تكون شبكة محلية أو عالمية، الهدف منها تشغيلهم كنظام واحد وهذا ما يعرف بالأنظمة الموزعة .

بالرغم من أن الأنظمة الموزعة تنجز أعمال المعالجة للبيانات الضخمة بشكل سريع . إلا أنه يوجد بها مشاكل تحتاج إلى حل أهمها : التنسيق والاتصال بين الأجهزة وبالعادة يتم اللجوء إلى كتابة أكواد تكون معقدة وطويلة لأداء هذه المهام مما يؤدي إلى زيادة الكود البرمجي .

وهذا السبب أدى إلى ظهور فكرة كل من الماب رديوس والزوكيبر، اللذان يقومان بعمليات التنسيق والاتصال بين الأجهزة دون تدخل المستخدم . بالتالي يتمكن المستخدم من التركيز والإهتمام بكتابة كوده البرمجي دون القلق بالعمليات الأخرى .

وفي هذا الفصل سوف نتعامل مع الهادوب الذي يتم من خلاله تنفيذ فكرة الماب رديوس وتوضيحة من خلال مثال عداد الكلمات (Word Count) .

3.1 أباتشي هادوب (Haddop)

هو برنامج أو منصه برمجية مفتوحة المصدر مكتوبة بلغة الجافا لتخزين ومعالجة البيانات الضخمة بشكل موزع مثل تخزين بيانات ضخمة على عدة أجهزة ومن ثم توزيع عملية المعالجة على هذه الاجهزة لتسريع نتيجة المعالجة.

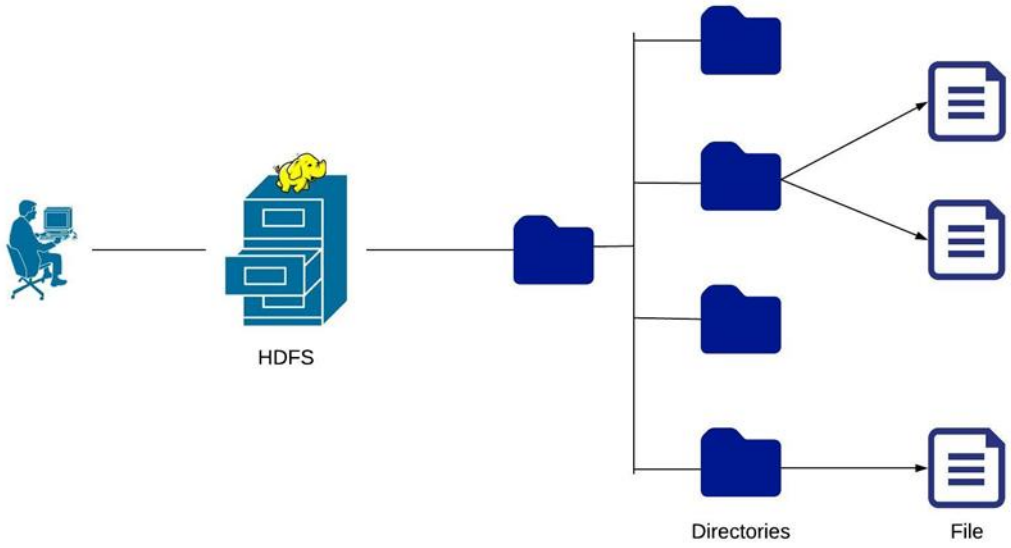
مكونات هادوب الأساسية

1. نظام ملفات هادوب الموزع (Hadoop Distributed File System (HDFS)) : يستخدم كنظام ملفات يوفر خدمات إنشاء وإدارة الملفات والمجلدات . والهدف الرئيسي منه هو تخزين الملفات الكبيرة.
2. الماب رديوس (MapReduce) : هي بيئة برمجية تستخدم المعالجة المتوازية في معالجة البيانات

3.1.1 نظام ملفات هادوب الموزع

يعتبر نظام ملفات هادوب الموزع نظام ملفات يوفر خدمات إنشاء وإدارة الملفات والمجلدات مثل نظم الملفات الموجوة في جميع نظم التشغيل العادية . فكل نظم التشغيل التي نتعامل معها تحتوي نظام ملفات يقوم بادارة الملفات ويوفر طريقة سهلة للتعامل مع الملفات وهو يقوم بالدور المعقد في التعامل مع التخزين مخفياً عنا التعقيدات التي تتم .

الشكل (10) يوضح نظام ملفات هادوب الموزع الذي يقدم خدمة إدارة الملفات للمستخدمين، و يتميز ببعض المميزات والخواص التي تجعله مختلف عن نظم الملفات العادية .



شكل 10:نظام ملفات هادوب الموزع

مميزات نظام ملفات هادوب الموزع

يتميز HDFS عن نظم الملفات التقليدية بمميزات منها :

1. موزع (distributed) : يمكن تخزين ملفات كبيرة جداً في نظام HDFS، فهو يوزع أجزاء الملفات في كل الاقراص في كل الاجهزة التي يود التعامل معها كأنها قرص واحد كبير.
2. قابل للتوسع (scalable) : كلما احتجت الى مساحة اكبر، ما عليك سوى ان تضيف اجهزة اضافية إلى التجمع .
3. تجاوز الأعطال (fault tolerant) : بما ان نظام HDFS يستخدم كثير من الاجهزة، فهذا يمكنه من الاستمرار في العمل حتى ولو تعطل البعض .

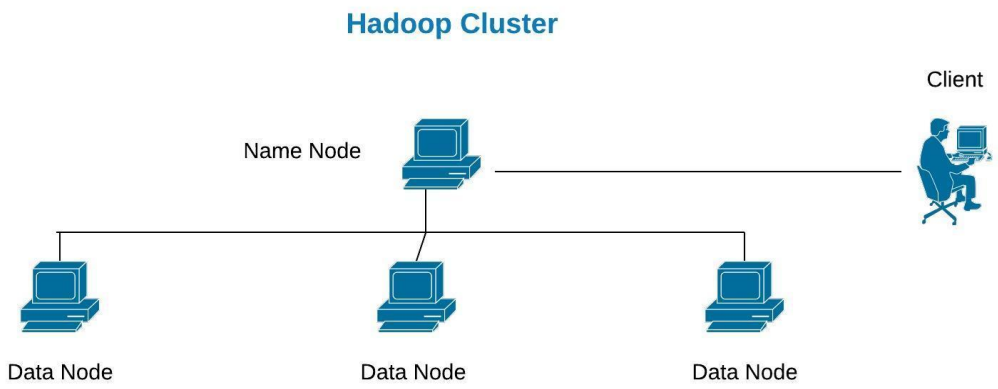
تجمع هادوب (Hadoop Cluster)

غالبا ما يتكون هدوب من مجموعة من الأجهزة الصغيرة (nodes) قليلة التكلفة المتوفره للمستخدمين العادين، ترتبط مع بعضها بشبكة سريعة، ويعمل نظام هادوب على أن يجعلها تظهر لمستخدميها كانها جهاز واحد كبير وسريع .ويخفي نظام هدوب تفاصيل التجمع عن المستخدم .

نلاحظ من الشكل (11) أن هذا التجمع يتكون من أربعة أجهزة, يحتوي على جهاز واحد رئيسي (master) ومجموعة من الأجهزة تكون خادمات (slave) . الجهاز الرئيسي يسمى Namenode وتسمى الخادمات Datanode .

مهمة الجهاز الرئيسي هي وصف البيانات (metadata), تخزين المدخلات,وإدارة اسماء المجلدات واسماء الملفات (manage file system namespace) . بينما تستخدم الخادمات في تخزين وإدارة ومعالجة البيانات. وللحفاظ على التواصل بين الأجهزة الرئيسية والخادمات ترسل الخادمات رسائل للأجهزة الرئيسية للتأكد بدورها من نشاط أو عمل الخادمات .

في التجمع, يتم توزيع البيانات على جميع الأجهزة الموجودة .وتقوم ملفات النظام (distributed file systems) بتقسيم البيانات الكبيرة إلى قطع بناءاً على عدد الخادمات أو الأجهزة .



شكل 11:تجمع أجهزة(cluster)

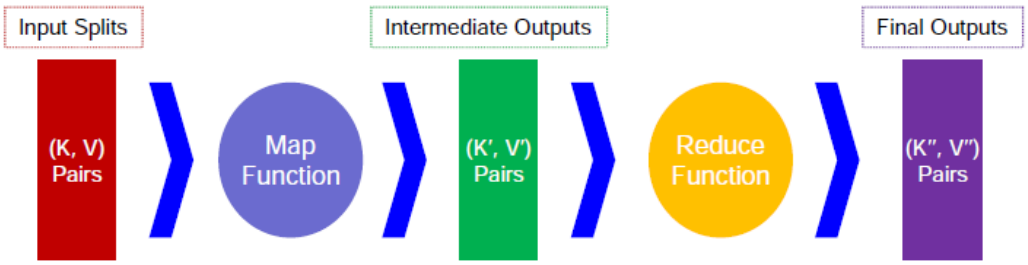
3.1.2 الماب رديوس

ماب رديوس هو نموذج برمجي مناسب لمعالجة البيانات الضخمة. هادوب قادر على تشغيل برامج الماب رديوس مكتوبة بلغات مختلفة منها: Java و Ruby و Python و C ++. برامج الماب رديوس متوازية في طبيعتها ، وبالتالي فهي مفيدة للغاية لإجراء وتحليل البيانات على نطاق واسع باستخدام أجهزة متعددة.

يعمل الماب رديوس على مرحلتين :

- مرحلة الماب (Map phase), مخرجات هذه المرحلة تسمى بالمخرجات الوسطية (intermediate outputs (IOs)). وذلك لإستهلاكها في مرحلة الرديوس .
- مرحلة الرديوس (Reduce phase), مخرجات هذه المرحلة تسمى بالمخرجات النهائية (final outputs (FOs)).

المدخلات و المخرجات لكل مرحلة من المراحل أعلاه هي عباره عن أزواج (key-value) موضحة في الشكل(12). بالإضافة إلى ذلك ، يحتاج كل مبرمج إلى تحديد وظيفتين : وظيفة الماب وظيفة الرديوس.



شكل 12:مدخلات ومخرجات الماب والرديوس[3]

3.1.2.1 مبدأ عمل الماب ريديوس

لنتمكن من فهم مبدأ عمل التجربة سوف نأخذ مثال بسيط يتكون من عدة أسطر يوضح طريقة العمل بسهولة.

على اعتبار أنه لدينا المدخلات التالية لبرنامج عداد الكلمات:

Welcome to PPU PPU ITCE ITCE Cloud Computing Welcome to PPU PPU Cloud Computing

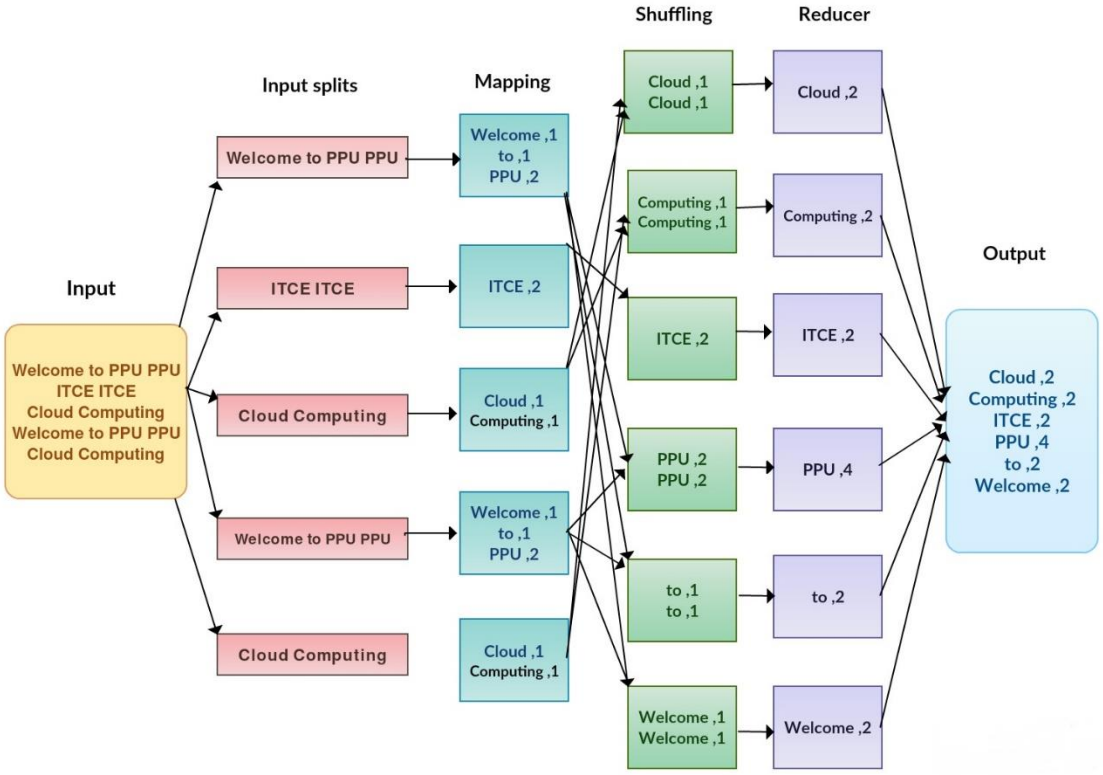
كما هو مبين بالشكل (13) تمر البيانات عبر المراحل التالية :

- تقسيم المدخلات (Inpur splits) :

يتم تقسيم المدخلات في ماب ريدوس إلى قطع ثابتة الحجم تسمى Input Splits, عادة يكون حجم القطعة الواحدة 64MB , تقسيم المدخلات هو جزء من المدخلات التي يستهلكها جزء mapping .

- Mapping :

هذه هي المرحلة الأولى في تنفيذ برنامج الماب ريديوس . في هذه المرحلة يتم تمرير البيانات في كل انقسام إلى map (تعبر عن الأجهزة) لإنتاج قيم المخرجات . في مثالنا، وظيفة الماب ريديوس هي حساب عدد مرات ظهور كل كلمة من انقسامات المدخلات وإعداد قائمة على شكل <key , value> .



شكل 13:مخطط لعمل ماب ريديوس

- Shuffling :

مدخلات هذه المرحلة هي مخرجات المرحلة السابقة . وتتمثل مهمتها في توحيد السجلات ذات الصلة الواحده من مخرجات المرحلة السابقة (تجمع key المتشابهين). في مثالنا، يتم تجميع الكلمات نفسها جنبا إلى جنب مع تكرارها .

- Reducing :

تجمع هذه المرحلة القيم من مرحلة السابقة وترجع قيمة إخراج واحد . وهي النتيجة النهائية.

3.1.2.1 كيف ينظم الماب رديوس العمل ؟

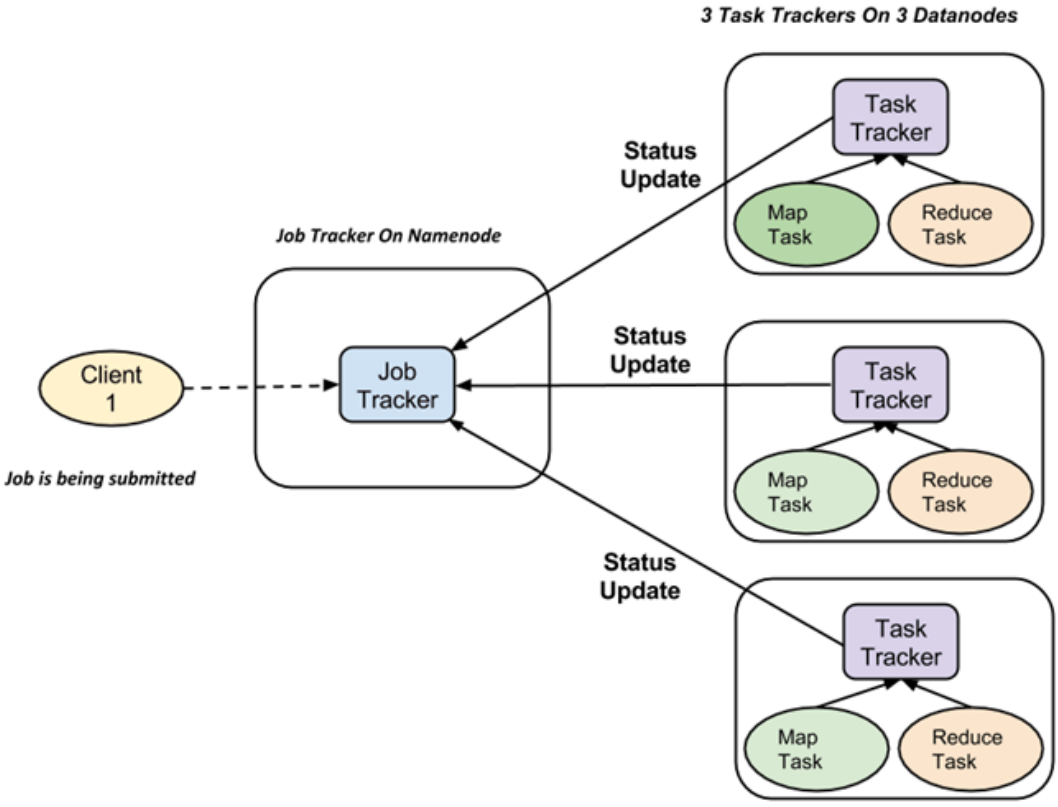
الهادوب يقسم المهمة (job) إلى مجموعة مهام (tasks). وهناك نوعان من المهام هما :

- مهام الماب (Spilts & Mapping)
- مهام الرديوس (Shuffling & Reducing)

يتم التحكم في عملية التنفيذ كاملة (مهام الماب ومهام الرديوس، كلاهما) بواسطة نوعين من البرامج يتعاونوا مع بعضهم لإتمام نجاح عملية المعالجة, هما:

1. JobTracker: هو البرنامج الموجود داخل الجهاز الرئيسي (master), والمسؤول عن تنفيذ المهمة.
2. Multiple TaskTrackers: هو البرنامج الموجود داخل الخادمات (slave), كل خادم يقوم بأداء مهمة واحدة ويراقب المهمة الخاصه به.

لكل مهمة يتم تقديمها للتنفيذ في النظام، يوجد jobtracker واحد موجود على Namenode وهناك العديد من TaskTrackers التي توجد على Datanode. موضح في الشكل(14).



شكل 14:تنظيم العمل في الماب رديوس[4]

بعض التفاصيل عن الشكل (14)

- تقسم المهمة إلى مهام متعددة يتم وضعها وتشغيلها بعد ذلك على عدد من datanodes في التجمع (cluster).
- تقع المسؤولية على Jobtracker للتنسيق عن طريق توزيع المهام على datanodes المختلفة.
- بعد ذلك يتم تنفيذ المهام بشكل فردي من خلال TaskTrackers، الذي يوجد على كل datanodes.
- مسؤولية TaskTrackers هي إرسال تقرير إلى Jobtracker.
- بالإضافة إلى ذلك ، يرسل TaskTrackers بشكل دوري رساله إلى Jobtracker لإعلامه بحالة النظام أو المهمة.
- وهكذا يتتبع Jobtracker التقدم العام لكل مهمة. في حالة فشل مهمة معينة ، يمكن ل jobtracker أن يعيد توزيع المهمة على TaskTrackers مختلف.

3.2 تجربة عداد الكلمات (Word Count)

3.2.1 وصف التجربة

تجربة عداد الكلمات تقوم بحسابة عدد مرات ظهور كل كلمة في مجموعة إدخال معينة (غالبا تكون كمية كبيرة من الوثائق), في تجربتنا تم اختيار كتاب ليتم حساب عدد ظهور كل كلمة من كلماته .

وتقوم التجربة على الماب ردیوس وهي مكون من مكونات الهادوب الرئيسية المذكوره أعلاه.

سوف تنفذ التجربة على جهاز واحد (single node) .وأيضاً على عدة أجهزة (multi node) ,الفرق هنا يكون في زيادة عدد الأوامر لإعدادات التجمع . أما بالنسبة لأوامر تجربة عداد الكلمات فهي نفسها . والنتيجة نفسها لكن عملية المعالجة سوف تكون على أكثر من جهاز .

وبدلا من الحديث عن تفاصيل لا حصر لها عن ماب ريديوس من الافضل ان نعرض تجربة عداد الكلمات التي تساعدنا في التعمق في فهم ماب ريديوس .

3.2.2 الهدف من التجربة

التعرف على الماب ردیوس الذي يقوم بمعالجة بيانات ضخمة على مجموعة من الأجهزة معاً دون القلق من عمليات الاتصال والتنسيق وتصحيح الأخطاء بين الأجهزة, وبالتالي تقليل الكود البرمجي .

3.2.3 المتطلبات اللازمة لتطبيق التجربة

تتمثل متطلبات التجربة كالتالي :

1. نظام تشغيل ابنتو (Linux Ubuntu 16.04), في التجربة تم الاعتماد على vmware لاحتضان الابنتو. وذلك لأن vmware يوجد بها اتصال بين أجهزة الأبنتو بداخله .
2. جافا بأي اصدار . في التجربة تم الاعتماد على الإصدار الثامن .
3. هادوب اصدار 2.6.5 .

3.2.4 تنفيذ التجربة

حتى يتمكن الطالب من تنفيذ هذه التجربة يجب أن يقوم أولا بتنصيب وإعداد هادوب من خلال مجموعه من الأوامر, ومن ثم تتبع أوامر تجربة الماب رديوس التي سوف يتم ذكرها لاحقاً .

ملاحظة :عند تنصيب هادوب يرفق معه العديد من المجلدات والملفات, من ضمنها المجلد : (hadoop-mapreduce-examples-2.6.5.jar) الذي يحتوي على (word count class) الموجود به أكواد كل من الماب والرديوس.

كود الماب

```
void Map (key, value){
        for each word x in value:
                output.collect(x, 1);
}
```

كود الرديوس

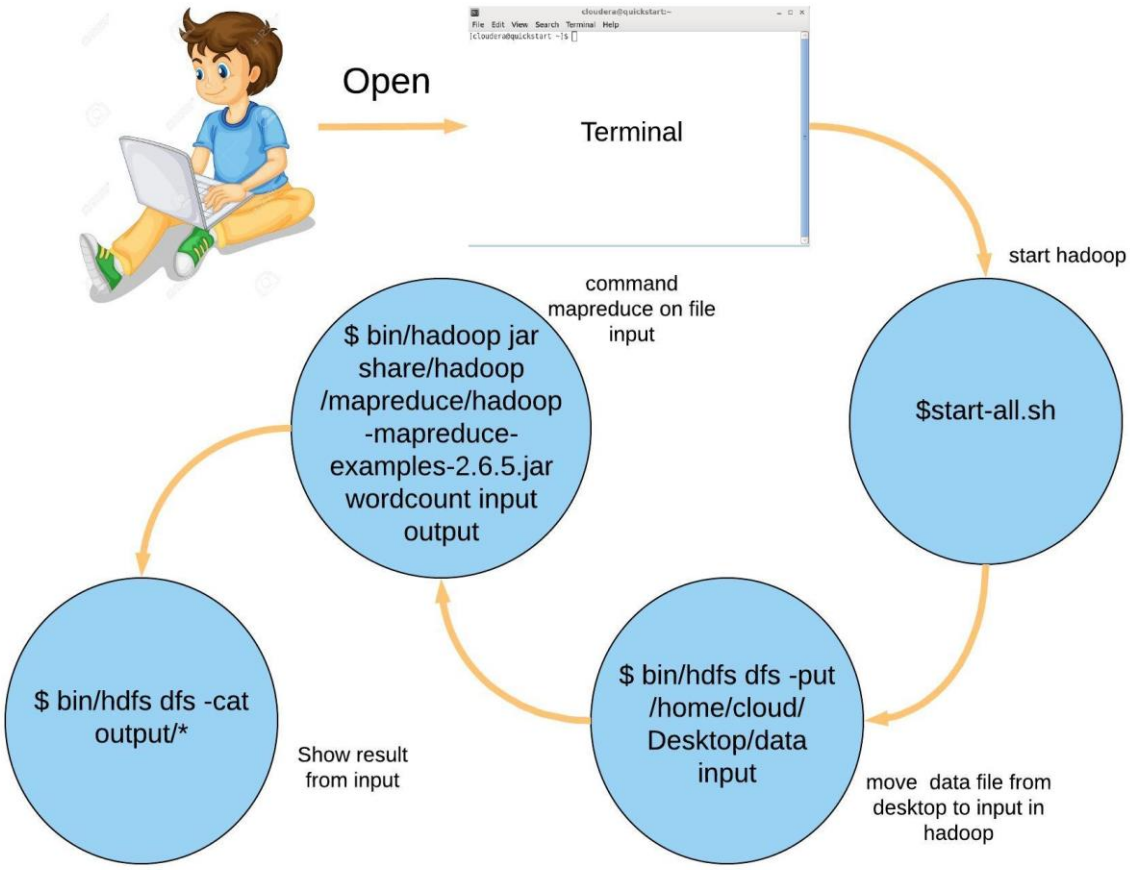
```
void Reduce (keyword, <list of value>){
        for each x in <list of value>:
                sum+=x;
        final_output.collect(keyword, sum);
}
```

امتداد المجلد (path) :

/usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.5.jar

3.2.5 أوامر التجربة

فيما يلي في الشكل (15) الأوامر الرئيسية لتجربة عدادا الكلمات التي تتم من خلال Terminal :



شكل 15:الأوامر الرئيسية لتجربة عداد الكلمات

الأوامر بشكل مفصل :

الأوامر المفصلة لتنصيب وإعدادات هادوب لجهاز واحد (single node)

أولا	
$ sudo apt-get update	تحديث للنظام من خلال تحديث لملف source list لتحديث التطبيقات الخاصة بالابنتو .
$sudo apt-get install default-jdk	تنزيل الإصدار الافتراضي للجافا غالبا يكون اخر اصدار عن طريق repository أو المستودع الخاص بالابنتو .
$ java –version	لفحص إصدار الجافا .
ثانياً	
$ sudo apt-get install ssh	تنزيل ssh اختصار ل Secure Shell يتكون من ssh الأمر الذي نستخدمه للاتصال بالأجهزة عن بعد. ومن sshd يعمل على الخادم ويسمح للعملاء بالاتصال بالخادم . (خاصة بالmultinde)
$ sudo apt-get install rsync	تنزيل rsync هي أداة مزامنة للشبكة.

الأوامر الخاصة بإعادات ssh. لأن هادوب يطلب الوصول إلى ssh من أجل إدارة الأجهزة الموجودة. وبالتالي نحتاج الى ssh يكون جاهز ومهيئ على الأجهزة .	
لإنشاء مفتاح من أجل المصداقية .	$ ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
يضيف المفتاح الذي تم إنشاؤه حديثًا إلى قائمة المفاتيح المصرح بها.	$ cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys
ثالثا	
تنزيل هادوب من الموقع الخاص به .	$ wget http://mirrors.sonic.net /apache/hadoop/common/hadoop-2.6.5/hadoop-2.6.5.tar.gz
فك ضغط الملف .	$ tar xvzf hadoop-2.6.5.tar.gz
نقل هادوب إلى ملفات النظام (directory).	$ sudo mv hadoop-2.6.5 /usr/local/hadoop
تحديث	$ update-alternatives --config java
رابعا	
إعدادت الملفات, يجب تعديل الملفات التالية لإكمال إعدادت هادوب .	
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64 export HADOOP_HOME=/usr/local/hadoop export PATH=$PATH:$HADOOP_HOME/bin export PATH=$PATH:$HADOOP_HOME/sbin export HADOOP_MAPRED_HOME=$HADOOP_HOME export HADOOP_COMMON_HOME=$HADOOP_HOME export HADOOP_HDFS_HOME=$HADOOP_HOME export YARN_HOME=$HADOOP_HOME export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib /native export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"	$ sudo gedit ~/.bashrc
ننتقل إلى المسار التالي من أجل إكمال تعديل الملفات الأخرى	$ cd /usr/local/hadoop/etc/hadoop
export JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64"	$ sudo gedit hadoop-env.sh
<property> <name>fs.defaultFS</name> <value>hdfs://localhost:9000</value> </property>	$ sudo gedit core-site.xml
<property> <name>yarn.nodemanager.aux-services</name> <value>mapreduce_shuffle</value> </property> <property> <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name> <value> org.apache.hadoop.mapred.ShuffleHandler</value> </property>	$ sudo gedit yarn-site.xml
لتغير اسم الملف الخاص بالmapred	$ sudo cp mapred-site.xml.template mapred-site.xml
<property> <name>mapreduce.framework.name</name> <value>yarn</value> </property>	$ sudo gedit mapred-site.xml

$ sudo gedit hdfs-site.xml	<property> <name>dfs.replication</name> <value>1</value> </property> <property> <name>dfs.namenode.name.dir</name> <value>file:/usr/local/hadoop/hadoop_data/hdfs/namenode</value> </property> <property> <name>dfs.datanode.data.dir</name> <value>file:/usr/local/hadoop/hadoop_store/hdfs/datanode</value> </property>
خامساً	
$ mkdir -p /usr/local/hadoop/ hadoop_data/hdfs/namenode	إنشاء ملف namenode
$ mkdir -p /usr/local/hadoop /hadoop_data/hdfs/datanode	إنشاء ملف datanose
$ sudo chown cloud:cloud -R /usr/local/hadoop	Chorn يغير ملكية الملفات في النظام.
سابعاً	
$ hdfs namenode -format	عمل فورمات لل filesystem لنتمكن من استخدام هادوب
$ start-all.sh	لتشغيل هادوب .
$ jps	لنتأكد من عمل كل من :NameNode DataNode NodeManager ResourceManager

جدول 3:إعدادات هادوب لجهاز واحد(single node)

الأوامر الإضافية لإعدادات التجمع (multi node)

سوف يكون لدينا جهازان على البيئة الافتراضية VMware, الأول hadoopmaster سوف يمثل namenode, الجهاز الثاني hadoopslave سوف يمثل datanode.

ملاحظة : يمكن زيادة عدد , hadoopslave لكن بسبب امكانيات الجهاز المحدودة لدينا سيكون هنا hadoopslave واحدة .

بعد إتمام إعدادات الهادوب لجهاز واحد أعلاه نكمل بالخطوات التالية :

أولا :داخل hadoopmaster	
Namenode > hadoopmaster > 192.168.144.130 Datanode > hadoopslave >192.168.144.135	نقوم بتحديد عناوين كل من master و slave
$ ifconfig	لنتمكن من معرفة عناوين الأجهزة نكتب الأمر التالي

$ sudo gedit /etc/hosts >>hadoopmaster 192.168.144.130 hadoopslave 192.168.144.135	لتعديل ملف hosts لجميع أجهزة (nodes) , من خلال تحديد عناوين الأجهزة بالإضافة إلى أسماءهم
$sudo gedit /etc/hostname >> hadoopmaster	لتعديل اسم hostname
الانتقال إلى المسار التالي داخل هادوب cd /usr/local/hadoop/etc/hadoop من أجل تعديل الملفات التالية	
$ sudo gedit core-site.xml	استبدال ال localhost إلى hadoopmaster
$ sudo gedit hdfs-site.xml	نستبدل القيمة 1 إلى 2 إذا كان لدينا 2datanode (تعبر عن عدد datanode)
$ sudo gedit yarn-site.xml	<property> <name>yarn.resourcemanager.resource-tracker.address</name> <value>hadoopmaster:8025</value> </property> <property> <name>yarn.resourcemanager.scheduler.address</name> <value>hadoopmaster:8030</value> </property> <property> <name>yarn.resourcemanager.address</name> <value>hadoopmaster:8050</value> </property>
$ sudo gedit mapred-site.xml	نستدل mapreduce.framework.name إلى mapred.job.tracker نستبدل yarn إلى hadoopmaster:54311
$ sudo rm -rf /usr/local/hadoop/hadoop_data/	لحذف جميع الملفات داخل hadoop_data
$ sudo gedit /usr/local/hadoop/etc/hadoop/masters >>hadoopmaster	لإضافة اسم hadoopmaster في نظام هادوب
$ sudo gedit /usr/local/hadoop/etc/hadoop/slaves >>hadoopslave1	لتعديل اسم hadoopslave في نظام هادوب بدل hostname
$ sudo gedit /usr/local/hadoop/etc/hadoop/hdfs-site.xml	لحذف datanode
$ sudo mkdir -p /usr/local/hadoop/hadoop_data/hdfs/namenode	لإنشاء directory namenode داخل الهادوب

ثانيا داخل hadoopslave	
$ sudo gedit /etc/hosts >>hadoopmaster 192.168.144.130 hadoopslave 192.168.144.135	لتعديل ملف hosts لجميع أجهزة (nodes) , من خلال تحديد عناوين الأجهزة بالإضافة إلى أسماءهم
$sudo gedit /etc/hostname >> hadoopslave	لتعديل اسم hostname
الانتقال إلى المسار التالي داخل هادوب cd /usr/local/hadoop/etc/hadoop من أجل تعديل الملفات التالية	
$ sudo gedit core-site.xml	استبدال ال localhost إلى hadoopmslave
$ sudo gedit hdfs-site.xml	نستبدل القيمة 1 إلى 2 إذا كان لدينا 2datanode (تعبر عن عدد datanode)
$ sudo gedit yarn-site.xml	<property> <name>yarn.resourcemanager.resource-tracker.address</name> <value>hadoopmaster:8025</value> </property> <property> <name>yarn.resourcemanager.scheduler.address</name> <value>hadoopmaster:8030</value> </property> <property> <name>yarn.resourcemanager.address</name> <value>hadoopmaster:8050</value> </property>
$ sudo gedit mapred-site.xml	نستدل mapreduce.framework.name إلى mapred.job.tracker نستبدل yarn إلى hadoopmaster:5431
$ sudo reboot	إعادة تشغيل النظام للوصول اليه عن بعد
$ sudo rm -rf /usr/local/hadoop/hadoop_data/	لحذف جميع الملفات داخل hadoop_data
$ sudo mkdir -p /usr/local/hadoop/hadoop_data/hdfs/datanode	لإنشاء directory datanode داخل الهادوب
$ sudo gedit /usr/local/hadoop/etc/hadoop/hdfs-site.xml	لحذف ال namenode منslave
ثالثا داخلhadoopmaster	
لإنشاء اتصال بين الأجهزة المختلفة $ sudo ssh-copy-id -i ~/.ssh/id_dsa.pub cloud@192.168.144.130	

$ sudo ssh-copy-id -i ~/.ssh/id_dsa.pub cloud@192.168.144.135
$ ssh 192.168.144.130
$ exit
$ ssh 192.168.144.135
$ exit	

$ hdfs namenode -format	عمل فورمات لل filesystem لنتمكن من استخدام هادوب
$ start-all.sh	لتشغيل هادوب .

جدول :4إعدادات هادوب لأكثر من جهاز (multi node)

أوامر تجربة عداد الكلمات :

يمكن تنفيذ أوامر التجربة التالية على جهاز واحد أو مجموعة أجهزة .

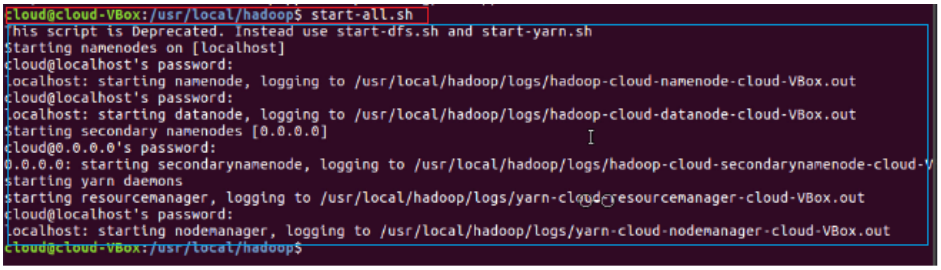
بعد إنشاء مجلد (data) وبه الملف (book.txt) الذي يحتوي على نص الكتاب المراد عمل الماب رديوس له نقوم بالخطوات التالية:

$ start-all.sh	لتشغيل هادوب .
$ cd /usr/local/hadoop	ننتقل إلى المسار داخل الهادوب
$ bin/hdfs dfs -mkdir /user	ننشئ ملف داخل directory وهو user
$ bin/hdfs dfs -mkdir /user/cloud	ننشئ داخل ال user ملف آخر ليكنcloud
$ bin/hdfs dfs -put /home/cloud/Desktop/data input	ننقل ملف data من سطح المكتب الى ملف input داخل مسار/user/cloud
$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.5.jar wordcount input output	الأمر الخاص بعمل ماب ريديوس على ملف inputوإظهار النتائج داخلoutput
$ bin/hdfs dfs -cat output/*	لإظهار النتائج من داخلoutput

جدول 5: أوامر تجربة عداد الكلمات

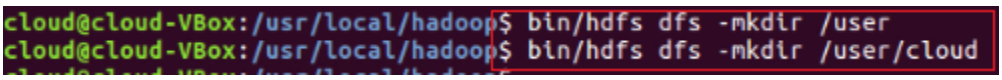
3.2.6 نتائج تنفيذ التجربة

الشكل (16) يوضح تشغيل هادوب



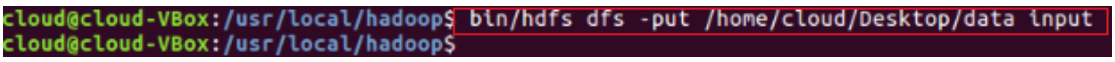
شكل 16:تشغيل هادوب

الشكل (17) يوضح إنشاء user&cloud



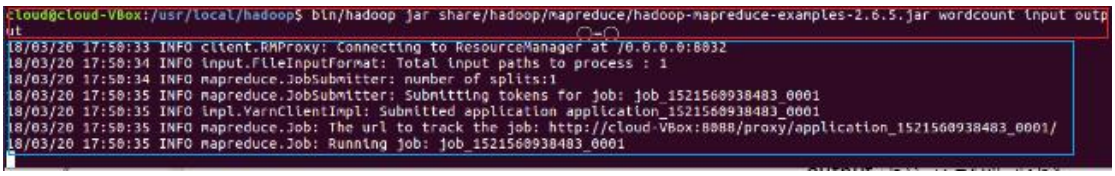
شكل 17:إنشاء user&cloud

الشكل (18) يوضح نقل ملف data من سطح المكتب الى ملف input داخل ملفات النظام ضمن المسار user/cloud/



شكل 18:نقل ملف data إلى input

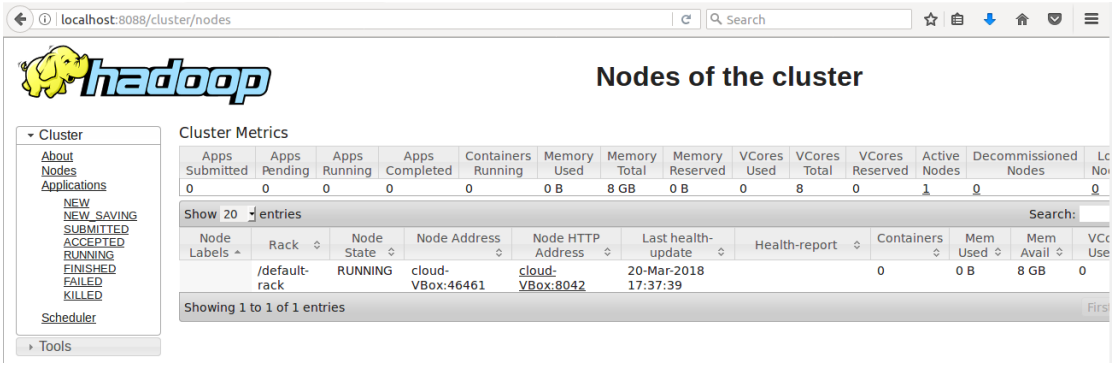
الشكل (19) يوضح عمل ماب ريديوس على ملف input الذي يحتوي على ملف الكتاب وإظهار النتائج داخل output



شكل 19:عمل الماب ريديوس

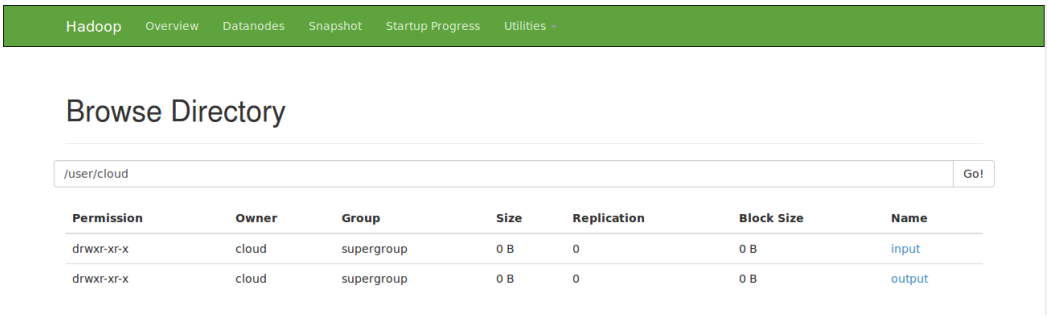
يمكن الوصول إلى البيانات داخل هادوب سواء قبل معالجتها أو بعد المعالجة والحصول على النتائج من خلال شاشات الويب, عن طريق وضع رقم المنفذ (port) داخل URL, ومنها :

1. localhost:8088 : للوصول إلى كافة التطبيقات التي توضح الأجهزة المتصلة, الشكل(20) يوضح الأجهزة المتصلة.

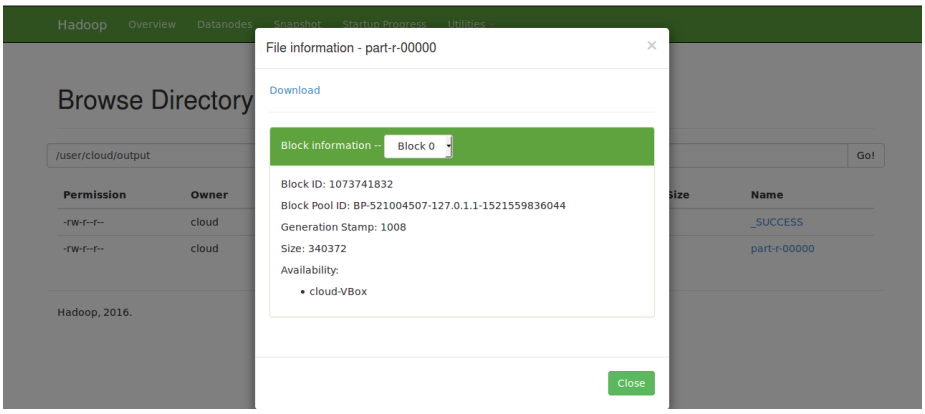


شكل 20:الأجهزة المتصلة داخل تجمع هادوب

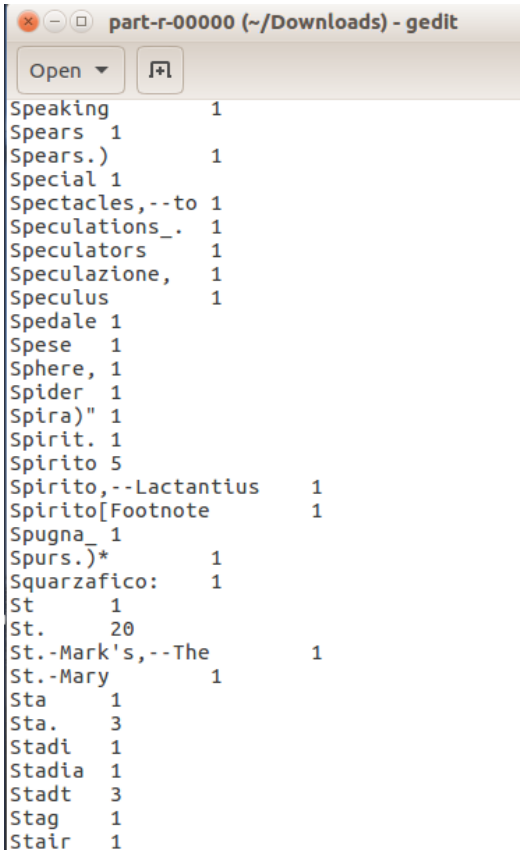
2. localhost:50070 : نستطيع رؤية جميع الملفات والمجلدات الخاصة بهادوب التي قمنا بانشاءها وتنزيلها على الجهاز. الشكل (21) والشكل (22) يوضح ذلك.



شكل 21:يظهر ملفات الإدخال والإخراج الخاصة بتجربة عداد الكلمات



شكل 22:داخل ملف الإخراج لنتمكن من تحميل النتائج من خلاله



شكل :23جزء من النتائج من الملف المحمل من خلال واجهة الويب

3.2.7 ملحقات التجربة

لقد تم تزويد هذه التجربة بالعديد من الملحقات التي تعتبر مساعدة للطالب حتى يتمكن من فهم هذه التجربة ، وتمثلت هذه الملحقات بالآتي :

الرابط الخاص بإعدادت الهادوب على جهاز واحد :

https://www.youtube.com/watch?v=4JkifuNwNz8&t=76s

الرابط الخاص بإعدادات هادوب على أكثر من جهاز :

https://www.youtube.com/watch?v=A5m3mXWZ3BA&t=8s

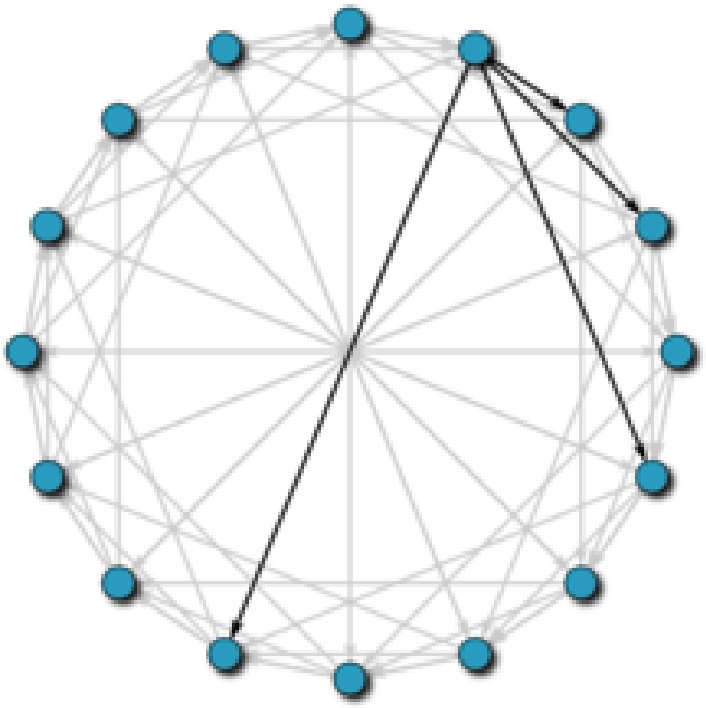
الرابط الخاص بتجربة عداد الكلمات :

https://www.youtube.com/watch?v=XQZRfzf15oQ&t=96s

الرابط الخاص بشرح مبسط للتجربة :

https://www.youtube.com/watch?v=pPFg5889FDE

الفصل الرابع: الجداول الموزعة(Distributed Hash Table (DHT))



الفصل الرابع: الجداول الموزعة ((Distributed Hash Table (DHT

في هذا الفصل سيتم التحدث عن موضوع الجداول الموزعة ومفهومها وخائصها, وأيضا التجربة الخاصة بها وكيفية تطبيقها.

4.1 مفهوم الجداول الموزعة

هي عبارة عن فئة من نظام موزع وغير مركزي توفر خدمات بحث مشابهة لجدول التجزئة (Hash Table) حيث تكون البيانات فيها مخزنة على شكل ازواج مكونة من قيمة ومفتاح {Key,Value} وكل جهاز (Node) مشارك يمكنه أن يستعيد القيمة الخاصة بمفتاح معين (ID) بكفاءة عالية ويتم توزيع مسؤولية الحفاظ على الأرتباط بين المفاتيح والقيم على كل الأجهزة المشاركة , حيث أن ذلك يؤدي الى تقليل نسبة التعطيل والتأخير في الحصول على الخدمة أو البيانات المطلوبة وهذا يساعد الجداول الموزعة على التوسع الى عدد كبير من الأجهزة وأيضا قدرة عالية في التعامل مع الوصول والمغادرة المستمرة للأجهزة وأيضا التعامل مع الأخطاء. وفيما يلي ذكر لخصائص الجداول الموزعة:

1. توزيع الأحمال (Load Balancing) : هي عبارة عن عملية توزيع أعباء العمل والمصادر الحاسوبية في بيئة الحوسبة السحابية .حيث أن هذه الخاصية تمكن من إدارة التطبيقات و توزيع المصادر المتوفرة على الحوسبة السحابية على جميع الأجهزة الموجودة .
2. قابلية التوسيع (Scalability) : هي قدرة النظام على الإستمرار بالعمل بشكل جيد عندما يحدث عليه تغيير سواءً بالحجم أو بالسعة الخاصة به وأيضا زيادة قدرة النظام على تحمل المزيد من البيانات وذلك حتى يتم تحقيق متطلبات المستخدم وأهدافه.
3. قابلية التوفر (Availability) : تشير هذه الخاصية الى إمكانية في الوصول الى المعلومات والمصادر من أي مكان وفي أي وقت . ويتم من خلالها ضمان ان البيانات ستكون متوفرة في الوقت المحدد وبالشكل المطلوب.
4. اللامركزية (Decentralization) : هي الخاصية التي تتمثل في تخصيص المصادر التي تتمثل بالأجهزة والبرامج لكل الاجهزة الموجودة وأيضا تدل على عدم وجود جهاز مركزي مسؤول عن كل الاجهزة , لكن يكون كل جهاز مسؤول عن نفسه .وتقوم هذه الأجهزة بمشاركة المصادر التي تكون موجودة على الشبكة.

سنركز فيما يلي على أحد أشهر الجداول الموزعة والذي يسمىChord DHT.

4.2 مقدمة عن Chord DHT

هو عبارة عن بروتوكول بحث موزع تم تطويره في جامعة MIT وتم نشره في مؤتمر Sigcomm للتكنولوجيا عام 2001. قام بحل مشكلة عامة وأساسية في شبكات الند للند (P2P). وهذه المشكلة تتمثل في كيفية تحديد موقع جهاز معين يعمل على تخزين بيانات معينة,وأيضا صممت لمواجهة التحديات التي تواجه أنظمة وتطبيقات الند للند (P2P) , ومن الأمثلة على هذه التحديات موازنة الأحمال , اللامركزية وإمكانية توفير البيانات والخدمات حتى لو كان هناك عطل أو مشكلة معينة.

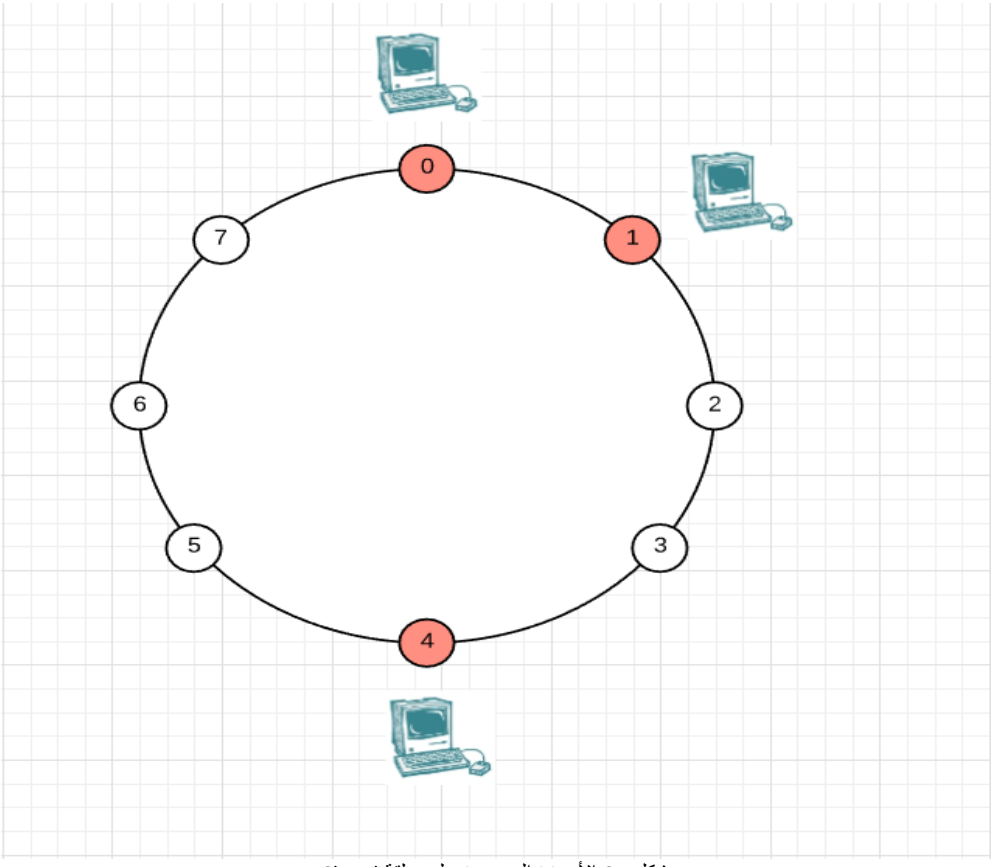
تكون الجداول الموزعة في Chord على شكل حلقة تسمى Chord ring وتركز Chord على كيفية تنظيم الأجهزة في شبكة تكون على شكل شبكة الند للند وتكون الأجهزة فيها مرتبطة بهذه الحلقة بناءً على اقتران محدد مسبقاً يسمى Hash function ويكون لهذه الأجهزة معرفات تسمى ID's يتم الحصول عليها من خلال ادخال عنوان الجهاز الى الإقتران حتى يمكن أن يتم وضعها على الحلقة بنفس الطريقة . تكون هذه المعلومات مخزنة على جدول يسمى Finger table حيث يعمل على تسريع عملية البحث عن الجهاز الذي يحتوي على البيانات المطلوبة.

فيما يلي سوف يتم التحدث عن نماذج Chord والتي تتمثل في نموذجين أساسيين هما النموذج البسيط والنموذج المحسن .

4.2.1 النموذج البسيط

هذا النموذج يستخدم Hash function يتم من خلاله الحصول على عناوين طولها 160 بت (M=160) . للحصول على معرف الجهاز (NodeID) يتم ادخال عنوان الجهاز (IP Address) لهذا الإقتران للحصول على عنوانه (ID) على Chord Ring. وأيضا يتم الحصول على عنوان الملف أو أي مصدر آخر (Object) على هذه الشبكة من خلال إدخال اسمه لهذا الإقتران. وكل جهاز يعرف ما هي الأجهزة التي تليه والتي تسبقه. ويتم وضع الملفات والمصادر الخاصة بالجهاز بين الجهاز نفسه والأجهزة التي تسبقه ,حيث أن كل جهاز يكون مسؤول عن الأجهزة التي تكون قبله. وأيضا تكون المسافة بين الأجهزة غير متماثلة وبإتجاه عقارب الساعة.

من خلال المثال التالي سنقوم بتوضيح العمليات الأساسية في Chord والتي تتمثل في إضافة أجهزة (Nodes) على الحلقة وعملية تخزين المصادر على الأجهزة الموجودة على الحلقة و أخيرا عملية استعادة المصادر المخزنة على تلك الأجهزة . في المثال يوجد حلقة Chord تحتوي على ثلاثة أجهزة وهي (0,1,4) . M=3 وهذا يعني أن عدد العناوين المحتمل الحصول عليها هو 2^3. الشكل (24) يوضح كيف سيكون شكل الحلقة وكيفية توزيع الاجهزة عليها .



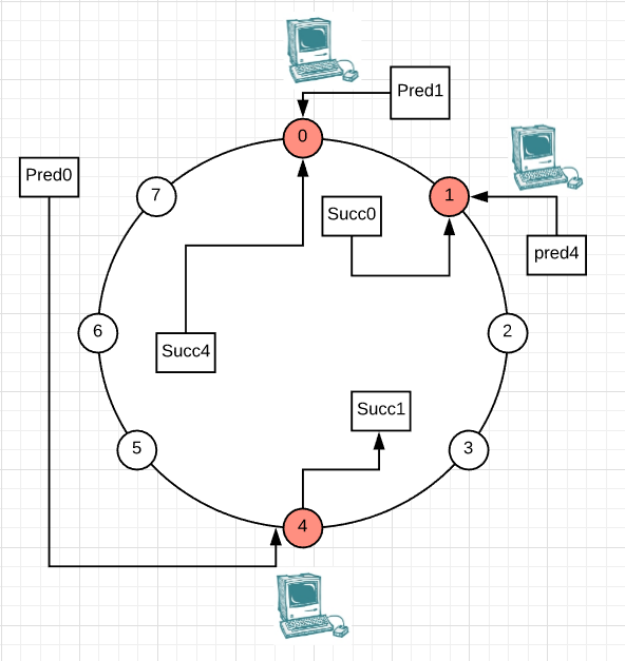
شكل 24:الأجهزة الموجودة على حلقة Chord

أولا: عملية إضافة جهاز إلى الحلقة (Join Node)

كل جهاز في هذه الحلقة يقوم بتسجيل معلومتين فقط وهما الجهاز الذي يليه ويسمى Successor والجهاز الذي يسبقه ويسمى Predecessor. وكل جهاز يكون مسؤول عن المنطقة التي تكون قبله. الشكل (25) يوضح الأجهزة التالية والسابقة للاجهزة الموجودة على الحلقة. والشكل (26) يوضح شكل Routing Table لهذه الأجهزة.

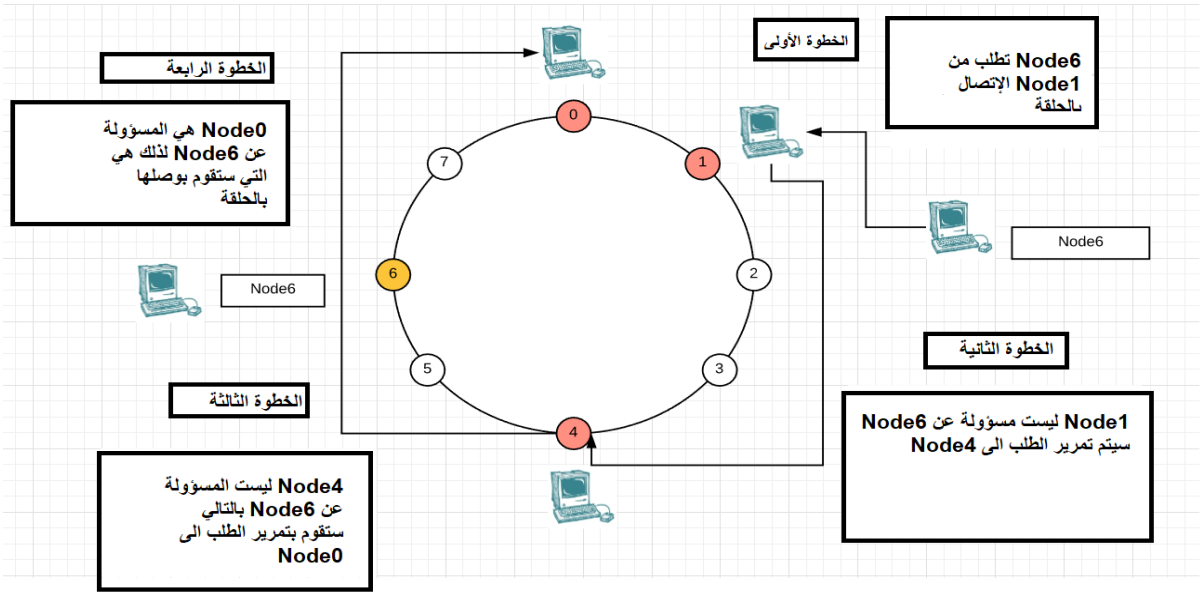
Node	Succ	Pred
Node0	Node1	Node4
Node1	Node4	Node0
Node4	Node0	Node1

شكل :25شكل Routing Table للأجهزة



شكل succ,pred:26 للأجهزة

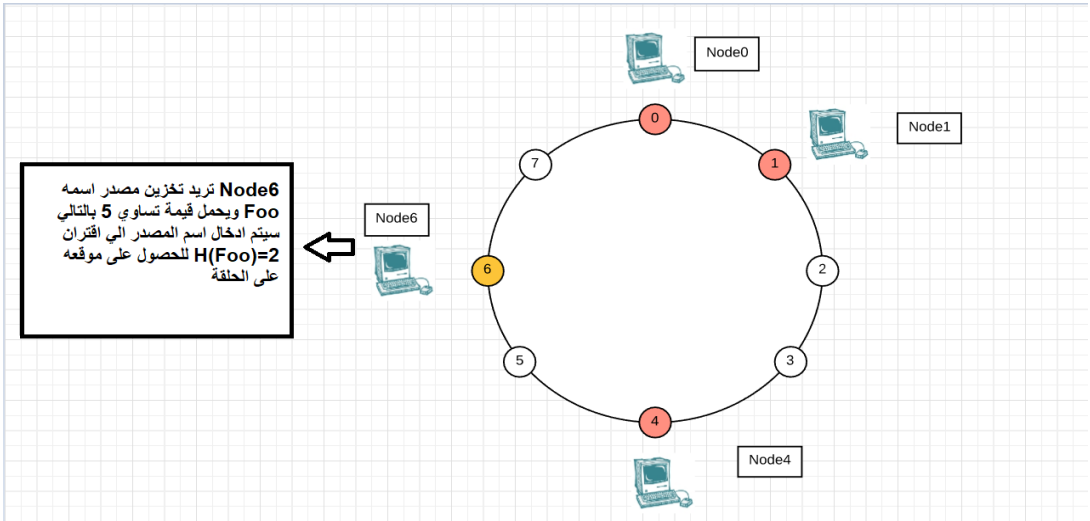
تتم عملية الأضافة كالتالي , يقوم الجهاز الجديد الذي يريد الإتصال بالحلقة بطلب الإتصال من أول جهاز يجده. إذا كان هذا الجهاز هو المسؤول عن الموقع الذي سيتصل به الجهاز الجديد سوف يستجيب لطلبه , أما إذا لم يكن هو المسؤول عنه سوف يتم تمرير الطلب الى الجهاز المسؤول عنه. الشكل (27) يوضح كيف تتم عملية إضافة Node6 على حلقة Chord .



شكل 27: عملية إضافة Node6

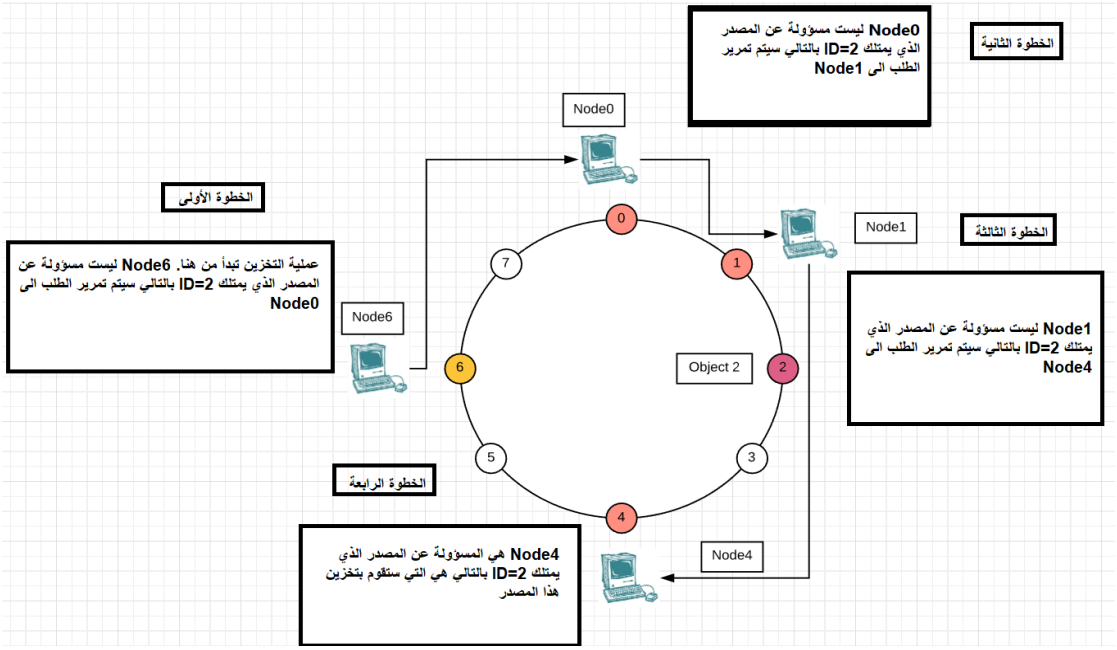
ثانيا: عملية تخزين المصادر على الأجهزة (Storing Objects)

عملية تخزين المصدر أو الملف على جهاز معين موجود على حلقة Chord تتم من خلال إدخال إسم المصدر أو الملف الى Hash Function وذلك حتى يتم الحصول على عنوانه على الحلقة . ويتم التخزين للمصدر أو الملف من قبل الجهاز الذي يكون مسؤول عن هذا العنوان . الشكل (28) يوضح الجهاز الذي يريد تخزين المصدر الذي إسمه Foo وقيمته هي 5 حيث يتم ادخال اسم هذا الملف الى Hash Function فنحصل على عنوانه على الحلقة وهو 2 .



شكل 28: الجهاز الذي يريد تخزين المصدر Foo

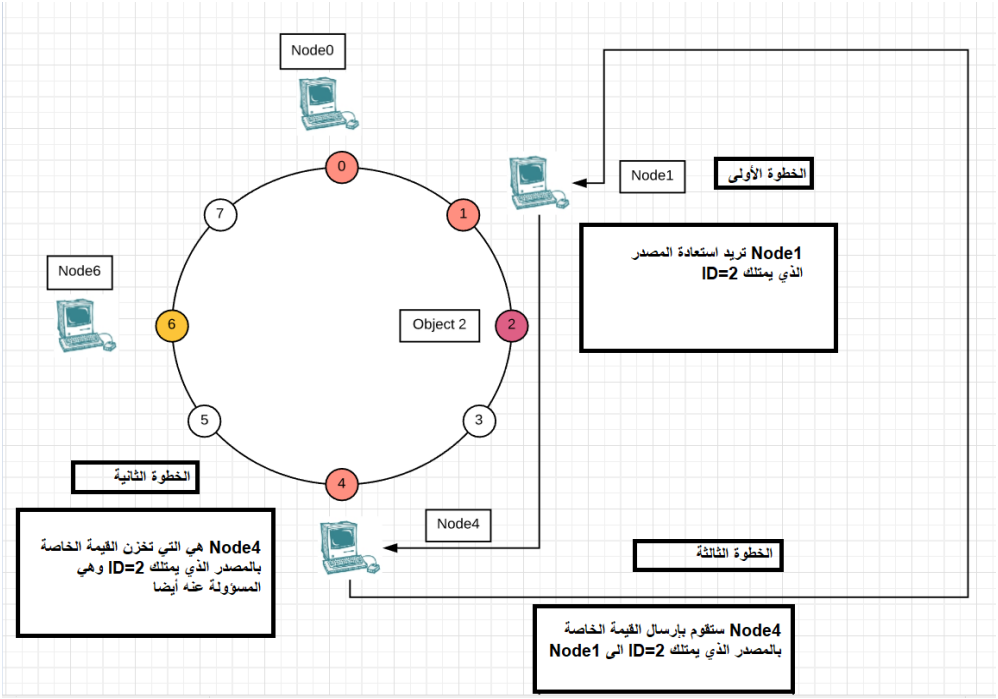
في الشكل (29) أدناه تم توضيح أن عملية التخزين للمصدر تبدأ عند الجهاز (6) و بما أنه ليس المسؤول عن Object2 تم تمريره إلى الجهاز (0) وبما أنه أيضا غير مسؤول عن Object2 سوف يتم تمريره الى الجهاز(1) لكنه غير مسؤول عن Object2 لذلك سيتم تمريره الى الجهاز (4) حيث أنه هو الجهاز المسؤول عن تخزين Object2 في الموقع 2 .



شكل 29: عملية التخزين للمصدر Foo

ثالثا: عملية استعادة المصادر التي تم تخزينها (Retrieving Objects)

في هذه العملية يقوم الجهاز الذي يريد استعادة مصدر معين (Object) بإرسال طلب الى أحد الأجهزة التي تكون موجودة على الحلقة. فإذا كان الجهاز الذي تلقى الطلب هو المسؤول عن هذا المصدر فإنه سيستجيب لهذا الطلب وسيقوم بإرجاع المصدر المطلوب الي الجهاز الذي قام بطلبه, أما إذا لا فسيتم تمرير هذا الطلب الى الجهاز المسؤول عنه. تتم هذه العملية من خلال قيام الجهاز الذي ارسل الطلب بإدخال اسم المصدر الى Hash Function للحصول على موقعه (ID) على الحلقة. ففي المثال يريد الجهاز (1) استعادة Object2 الذي اسمه Foo. قام الجهاز (1) بإرسال طلب الى الجهاز (4) وهو الجهاز المسؤول عن المصدر Object2 وايضا مسؤول عن تخزين القيمة الخاصة بالمصدر Object2 بتالي سيقوم بإرجاعها الى الجهاز (1).
الشكل (30) أدناه يوضح هذه العملية.



شكل 30: عملية استعادة قيمة Object2

هنا نكون قد انتهينا من شرح نموذج Chord البسيط. فيما يلي سيتم شرح النموذج المحسن من Chord DHT.

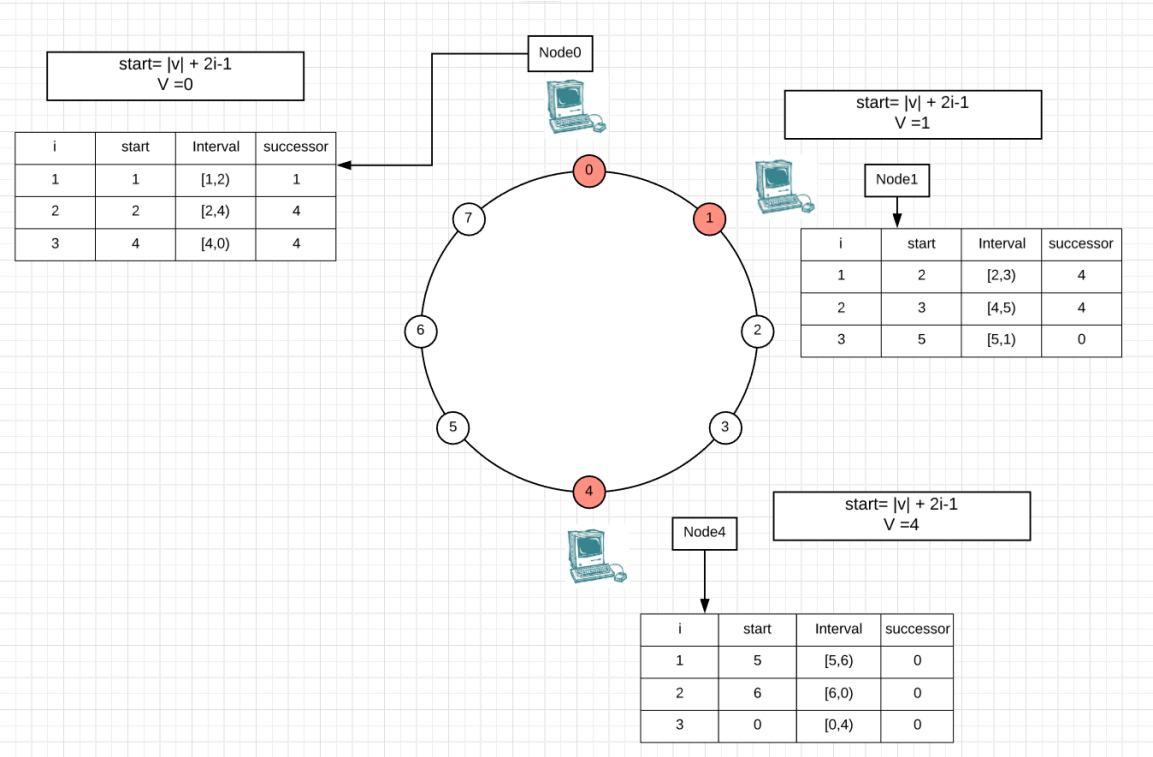
4.2.2 النموذج المحسن (Improved Chord)

يعتبر هذا النموذج افضل بكثير من نموذج Chord البسيط حيث يوفر عملية توجيه (Routing) وتعتبر هذه العملية أكثر قابلية للتوسع والسبب في ذلك هو وجود جدول خاص لكل الأجهزة الموجودة على الحلقة وهذا الجدول يسمى Finger Table حيث يحتوي هذا الجدول على معلومات عن الاجهزة الأخرى التي تكون موجودة على الحلقة وهذا يؤدي إلى تسريع وتحسين آداء عملية التوجيه, وأيضا يسهل عملية معرفة الاجهزة التي يمكن سؤالها عن معلومات معينة.

طريقة انشاء جدول Finger Table تكون كالتالي :

1. تحديد عدد المدخلات التي تكون خاصة بكل جهاز والتي يتم استنتاجها من قيمة M.
2. حساب البداية (Start) لكل مؤشر من خلال المعادلة التالية: $Start = 2^{(i-1)} + |V|$. حيث أن V تدل على رقم الجهاز الذي يتم عمل الجدول له.
3. تحديد الفترة (Interval) الخاصة بكل مؤشر في هذا الجدول
4. تحديد الجهاز التالي (Successor) الخاص بهذا المؤشر.

الشكل (31) أدناه يوضح شكل Finger Table للأجهزة الموجودة على الحلقة .حيث ان قيمة M=3 وبالتالي فإن عدد المدخلات لهذا الجدول هو 3 ويحتوي على 3 معلومات أساسية هي Start , Interval , Successor.



شكل 31:شكل Finger Table للأجهزة

4.3 تجربة Chord Distributed Hash Table

4.3.1 وصف التجربة

تجربة Chord DHT مأخوذة من موقع GitHub وتم التعديل على أكواد هذه التجربة بحيث تصبح اكثر قابلية للإستيعاب من قبل الطلاب وذلك من خلال إضافة تعليقات توضيحية للطلاب على الأكواد الخاصة بهذه التجربة وأيضا التعديل على شكل الناتج لإبرازه عند تنفيذها. هذا هو الرابط الخاص بالموقع الذي تم إحضار هذه التجربة منه https://github.com/mitul227/Chord-DHT.

تحتوي هذه التجربة على مجموعة من الملفات الاساسية وهي :

1. Main.cpp : هو الملف الذي يتم من خلاله تشغيل التجربة.
2. Function.cpp : هو الملف الذي يحتوي على كل الإقترانات المهمة للتجربة.
3. Class NodeInformation : يحتوي على المعلومات الخاصة بكل الاجهزة التي تكون موجودة على الاحلقة ويحتوي على كل الإقترانات التي تقوم هذه الأجهزة بتطبيقها للحفاظ على المعلومات الصحيحة عن الحلقة.
4. Class SocketAndPort : يحتوي على معلومات عن المدخل (Socket) , المنفذ (Port) والعنوان (IP Address) الخاص بالاجهزة
5. Class HelperClass : يحتوي على اقترانات مساعدة يتم طلبها لتنفيذ مهام معينة.

4.3.2 الهدف من التجربة

الهدف الأساسي من التجربة هو توصيل مفهوم الجداول الموزعة للطلاب وتوضيح مبدأ عملها لهم, وتوضيح كيف تقوم هذه الجداول بتسريع عملية التخزين و البحث عن المعلومات والوصول اليها.

4.3.3 المتطلبات اللازمة لتطبيق التجربة

تتمثل متطلبات التجربة بالآتي:

- برنامج لإستعراض الأكواد الخاصة بالتجربة مثل Visual Studio Code أو Sublime
- يمكن تطبيق هذه التجربة على أي نظام تشغيل. في مشروعنا قمنا بتطبيقها على نظام تشغيل لينوكس (Linux Ubuntu).

4.3.4 تنفيذ التجربة

حتى يتمكن الطالب من تنفيذ هذه التجربة يوجد أمران مهمان يتم من خلالهما تشغيل التجربة.الأمر الأول هو make حيث أنه يعتبر أداة يمكن من خلالها تحديد أي جزء من البرنامج سوف يتم عمل إعادة ترجمة له برمجياً (Recompilation) بشكل أوتوماتيكي, فعندما يتم كتابة هذا الأمر سوف يتم عمل إعادة ترجمة للجزء الذي تم

تحديده وفي هذه التجربة الجزء الذي سوف يتم عمل إعادة ترجمة له هو main.cpp. الأمر الثاني هو prog/. الذي يقوم بفتح Shell خاصة يتم كتابة الاوامر الخاصة بالتجربة فيها.

4.3.5 أوامر التجربة

فيما يلي ذِكْر للأوامر التي سيتم تنفيذها خلال هذه التجربة:

1. Create : الأمر الذي يتم من خلاله انشاء الحلقة ووضع جهاز أساسي عليها
2. Printstate : الأمر الذي يتم من خلاله طباعة المعلومات الخاصة بالأجهزة الموجودة على الحلقة وتتضمن هذه المعلومات ما يلي Predecessors, Successor List, Finger Table .
3. Print : الامر الذي يقوم بطباعة كل المفاتيح والقيم الموجودة على الجهاز الحالي او على جهاز معين
4. Port : الأمر الذي يتم من خلاله طباعة رقم المنفذ الذي يستمع لهالجهاز المرتبط بالحلقة .
5. Port Number :الأمر الذي يقوم بتغيير رقم المنفذ الذي يستمع اليه جهاز معين.
6. Join IP Port : الأمر الذي يتم من خلاله إضافة جهاز جديد إلى الحلقة من خلال إعطائه عنوان ومنفذ الجهاز الأساسي الموجود على الحلقة.
7. Put key value : الأمر الذي يتم من خلاله تخزين البيانات على الأجهزة الموجودة على الحلقة.
8. Get Key : الأمر الذي يتم من خلاله استرجاع بيانات تكون مخزنة على الأجهزة المتصلة على الحلقة.

4.3.6 نتائج تنفيذ التجربة

فيما يلي توضيح للنتائج التي ظهرت عند تنفيذ الاوامر الخاصة بالتجربة:

1. عمل إعادة ترجمة لها: تتم هذه العملية من خلال الأمر make. الشكل (32) توضح النتيجة من تنفيذ هذا الأمر.

```
ubuntu@ubuntu-VirtualBox:~/Documents/Chord-DHT-master$ make
g++ main.o init.o functions.o port.o nodeInformation.o helperClass.o -o prog -lc
rypto -lpthread
```

شكل 32:تنفيذ الأمر make

2. فتح Shell خاص بالتجربة لتنفيذ الأوامر الخاصة بها وتتم هذه العملية من خلال الأمر prog/. الشكل (33) توضح نتيجة تنفيذ هذا الأمر.

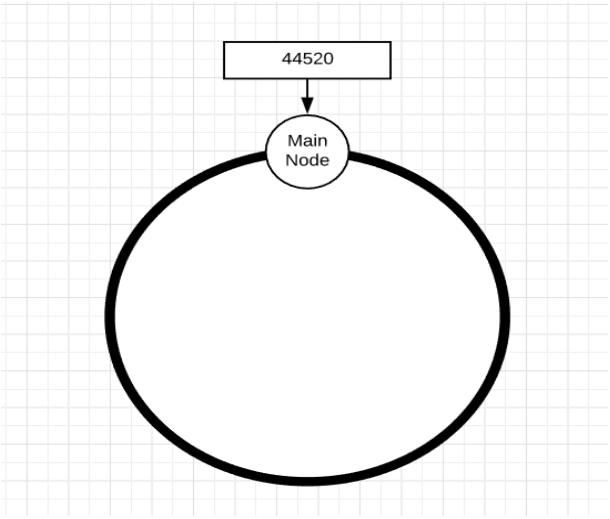
```
ubuntu@ubuntu-VirtualBox:~/Documents/Chord-DHT-master$ ./prog
Now listening at port number 44520
Type help to know more
>
```

شكل 33:نتيجة تنفيذ الأمر prog/.

3. إنشاء حلقة Chord وتتم هذه العملية من خلال الأمر create. الصورة (34) توضح نتيجة تنفيذ هذا الأمر . الشكل (35) توضح شكل الحلقة عند انشائها ,حيث أنه فور انشائها سيتم وضع الجهاز الأساسي عليها.

```
ubuntu@ubuntu-VirtualBox:~/Documents/Chord-DHT-master$ ./prog
Now listening at port number 44520
Type help to know more
> create
>
```

شكل 34:نتيجة تنفيذ الامر create



شكل :35حلقة Chord عند انشائها

4. طباعة المعلومات الخاصة بالجهاز الرئيسي الموجود على الحلقة والتي تتمثل في كل من Predecessor, Sucessor , SuccessorList ,Finger Table , وهذه العملية تتم من خلال الأمر printstate. الأشكال (36 , 37) توضح نتيجة تنفيذ هذا الأمر.

```
> printstate
Self 127.0.0.1 44520 210866615635109
Succ 127.0.0.1 44520 210866615635109
Pred 127.0.0.1 44520 210866615635109
****************
The Finger Table
****************
Fingers             ID's                 IP's          Ports
Finger[1] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[2] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[3] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[4] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[5] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[6] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[7] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[8] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[9] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[10] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[11] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[12] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[13] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[14] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[15] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[16] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[17] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[18] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[19] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[20] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[21] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[22] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[23] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[24] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[25] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[26] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[27] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[28] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[29] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[30] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[31] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[32] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[33] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[34] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[35] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[36] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[37] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[38] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[39] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[40] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[41] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[42] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[43] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[44] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[45] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[46] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[47] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
Finger[48] : ID :[210866615635109] : IP :[127.0.0.1] : Port :[44520]
```

شكل 36:معلومات الجهاز الرئيسي والتي تتمثل في Finger Table

```
******************
The Successors List
******************
Successors        SuccessorsID's          SuccessorIP's      successorsPorts
Successor[1] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[2] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[3] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[4] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[5] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[6] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[7] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[8] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[9] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[10] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
```

شكل 37:معلومات الجهاز الرئيسي والتي تتمثل في SuccessorList

5. إضافة أجهزة جديدة إلى الحلقة وهذه العملية تتم من خلال الأمر join ip port. الأشكال (38,39 , 40) توضح اضافة ثلاثة أجهزة جديدة وهي Node3 64808 , Node2 29812 , Node1 26433 . الشكل (41) توضح شكل الحلقة بعد إضافة هذه الأجهزة اليها.

```
ubuntu@ubuntu-VirtualBox:~/Documents/Chord-DHT-master$ ./prog
Now listening at port number 26433
Type help to know more
> join 127.0.0.1 44520
Successfully joined the ring
>
```

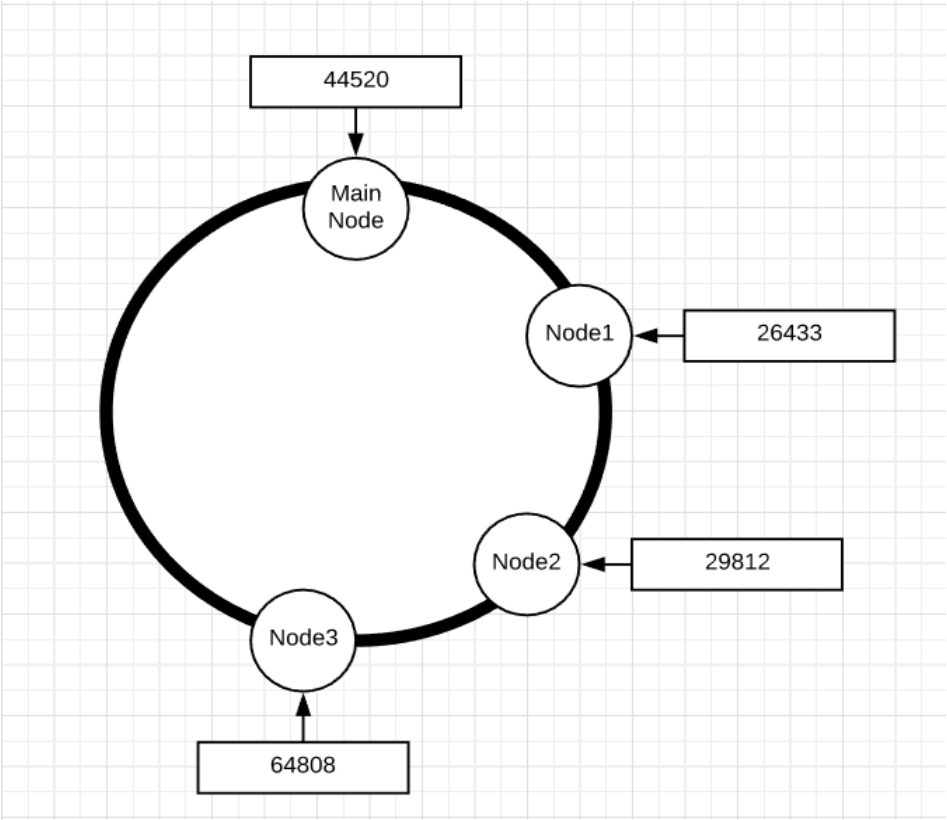
شكل 38:عملية إضافة الجهاز 26433

```
ubuntu@ubuntu-VirtualBox:~/Documents/Chord-DHT-master$ ./prog
Now listening at port number 64808
Type help to know more
> join 127.0.0.1 44520
Successfully joined the ring
>
```

شكل 39:عملية إضافة الجهاز 64808

```
ubuntu@ubuntu-VirtualBox:~/Documents/Chord-DHT-master$ ./prog
Now listening at port number 29812
Type help to know more
> join 127.0.0.1 44520
Successfully joined the ring
>
```

شكل 40:عملية إضافة الجهاز 29812



شكل 41:شكل الحلقة بعد إضافة الاجهزة الجديدة

6. تخزين معلومات على الأجهزة الموجودة على الحلقة , وهذه العملية تتم من خلال الأمر put key value. الأشكال (44,43,42) توضح نتائج تنفيذ هذا الأمر.

```
> put 11 123
*****************************************
Key is 11 and hash : 89261281685184
*****************************************
key entered successfully
>
```

شكل 42:تخزين القيمة 123

```
> put 12 456
*****************************************
Key is 12 and hash : 63428495609290
*****************************************
key entered successfully
>
```

شكل 43:تخزين القيمة 456

```
> put 13 789
****************************************
Key is 13 and hash : 976288601572
****************************************
key entered successfully
>
```

شكل 44:تخزين القيمة 789

7. استعادة البيانات من الاجهزة الموجودة على الحلقة , وهذه العملية تتم من خلال الأمر get key وبما أن كل جهاز يكون لديه علم بالأجهزة الاخرى التي تكون موجودة على الحلقة يمكن أن يتم الوصول للمعلومات المخزنة من قبل كل الأجهزة الموجودة على الحلقة. الأشكال (47,46,45) توضح نتائج تنفيذ هذا الأمر.

```
> get 12
********************************************
Found 12 : 456
********************************************
> get 13
********************************************
Found 13 : 789
********************************************
>
```

شكل 45:استعادة البيانات من قبل الجهاز 26433

```
> get 11
********************************************
Found 11 : 123
********************************************
> get 13
********************************************
Found 13 : 789
********************************************
>
```

شكل 46:استعادة البيانات من قبل الجهاز 64808

```
> get 11
********************************************
Found 11 : 123
********************************************
> get 12
********************************************
Found 12 : 456
********************************************
>
```

شكل 47:استعادة البيانات من قبل الجهاز 29812

8. طباعة المعلومات الخاصة بالاجهزة مرة أخرى من خلال الأمر printsatet. سنجد أن بعض المعلومات قد تغيرت بعد إضافة الاجهزة الجديدة الى الحلقة .الصور (49,47) توضح نتائج تنفيذ هذا الامر.

```
> printstate
Self 127.0.0.1 44520 210866615635109
Succ 127.0.0.1 64808 20068149825667
Pred 127.0.0.1 29812 201977274006634
****************
The Finger Table
****************
Fingers             ID's                 IP's         Ports
Finger[1] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[2] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[3] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[4] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[5] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[6] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[7] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[8] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[9] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[10] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[11] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[12] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[13] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[14] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[15] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[16] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[17] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[18] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[19] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[20] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[21] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[22] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[23] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[24] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[25] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[26] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[27] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[28] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[29] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[30] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[31] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[32] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[33] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[34] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[35] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[36] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[37] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[38] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[39] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[40] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[41] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[42] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[43] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[44] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[45] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[46] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[47] : ID :[20068149825667] : IP :[127.0.0.1] : Port :[64808]
Finger[48] : ID :[201977274006634] : IP :[127.0.0.1] : Port :[29812]
```

شكل 48:المعلومات التي تغيرت بعد إضافة الأجهزة الجديدة

```
******************
The Successors List
******************
Successors        SuccessorsID's          SuccessorIP's      successorsPorts
Successor[1] SuccID :[20068149825667] SuccIP :[127.0.0.1] SuccPort :[64808]
Successor[2] SuccID :[54196343614263] SuccIP :[127.0.0.1] SuccPort :[26433]
Successor[3] SuccID :[201977274006634] SuccIP :[127.0.0.1] SuccPort :[29812]
Successor[4] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[5] SuccID :[20068149825667] SuccIP :[127.0.0.1] SuccPort :[64808]
Successor[6] SuccID :[54196343614263] SuccIP :[127.0.0.1] SuccPort :[26433]
Successor[7] SuccID :[201977274006634] SuccIP :[127.0.0.1] SuccPort :[29812]
Successor[8] SuccID :[210866615635109] SuccIP :[127.0.0.1] SuccPort :[44520]
Successor[9] SuccID :[20068149825667] SuccIP :[127.0.0.1] SuccPort :[64808]
Successor[10] SuccID :[54196343614263] SuccIP :[127.0.0.1] SuccPort :[26433]
```

شكل 49:المعلومات التي تغيرت في SuccessorList

4.3.7 ملحقات التجربة

لقد تم تزويد هذه التجربة بالعديد من الملحقات التي تعتبر مساعِدة للطالب حتى يتمكن من فهم هذه التجربة, وتمثلت هذه الملحقات بالآتي:

- فيديو توضيحي بإستخدام الرسوم المتحركة (Animation video)
https://www.youtube.com/watch?v=0cdrLk0LTGE&t=15s

- فيديو خاص بإعدادات وتنفيذ التجربة (Execution video)
https://www.youtube.com/watch?v=M2-rWYU4SHQ&t=6s

- الملفات الخاصة بتنفيذ التجربة
https://github.com/amronoor/Chord-Distributed-Hash-Table

الفصل الخامس: كاسندرا(Cloud Storage /Apache Cassandra)

الفصل الخامس: كاساندرا (Cloud Storage /Apache Cassandra)

في هذا الفصل سوف يتم التحدث عن إحدى طرق التخزين على السحابة وتوضيح مفهومها وخصائصها وأيضا توضيح كيفية عملها والتجربة الخاصة بها.

5.1 مفهوم التخزين السحابي (Cloud storage)

هو عبارة عن خدمة توفر إمكانية الحفاظ على البيانات وإدارتها وعمل نُسخ إحتياطية لها أيضا, وتعمل على زيادة إمكانية توفر هذه البيانات للمستخدمين عبر الشبكة. يقوم المستخدم بالحصول على هذه الخدمة من خلال قيامه بدفع مبلغ مالي يتناسب مع السعة التخزينية التي يريدها خلال فترة زمنية معينة.

5.2 مفهوم قواعد البيانات الغير علائقية (NoSql Database)

هي عبارة عن قواعد بيانات توفر طريقة و آلية لتخزين و استرجاع المعلومات, وهذا النوع من قواعد البيانات مصممة لإستعياب مجموعة واسعة من البيانات وأيضا اشكال مختلفة منها.ويعتبر هذا النوع من قواعد البيانات بديلاً لقواعد البيانات العلائقية التقليدية. يتم إستخدام هذا النوع للعمل مع كميات كبيرة من البيانات التي تكون موزعة.وخلال هذا الفصل سوف يتم التحدث عن قاعدة البيانات الغير علائقية التي تسمى كساندرا والتي سيتم استخدامها للتخزين على السحابة.

5.3 مفهوم كاساندرا (Cassandra)

هي عبارة عن قاعدة بيانات موزعة ومفتوحة المصدر مصممة لتخزين وعمل إدارة لكميات ضخمة من البيانات بين مجموعة من الخوادم, وتعتبر Column oriented database حيث يتم تخزين البيانات فيها على شكل أعمدة وليس على شكل صفوف. في البداية كانت عبارة عن مشروع مشترك بين شركة جوجل (google) وشركة ياهو (Yahoo), لكن بعد ذلك انتقل الي شركة فيس بوك (Facebook) حيث أنها صممت ليكون الإتصال بين الأجهزة على شبكة الند للند (Peer 2 Peer) بدلاً من أن يكون هناك جهاز رئيسي وكل الأجهزة الأخرى متصلة به وذلك حتى يتم تفادي مشكلة الفشل في الجهاز الرئيسي (Single point of failure).وبعدما قامت شركة فيس بوك بفتح المصدر الخاص بكاساندرا أصبحت مشروع لشركة Apache. ومن الأمثلة على الشركات التي تستخدم كاساندرا شركة تويتر(Twiter), سيسكو (Cisco) وغيرها.وفيما يلي ذكر لبعض خصائص كاساندرا:

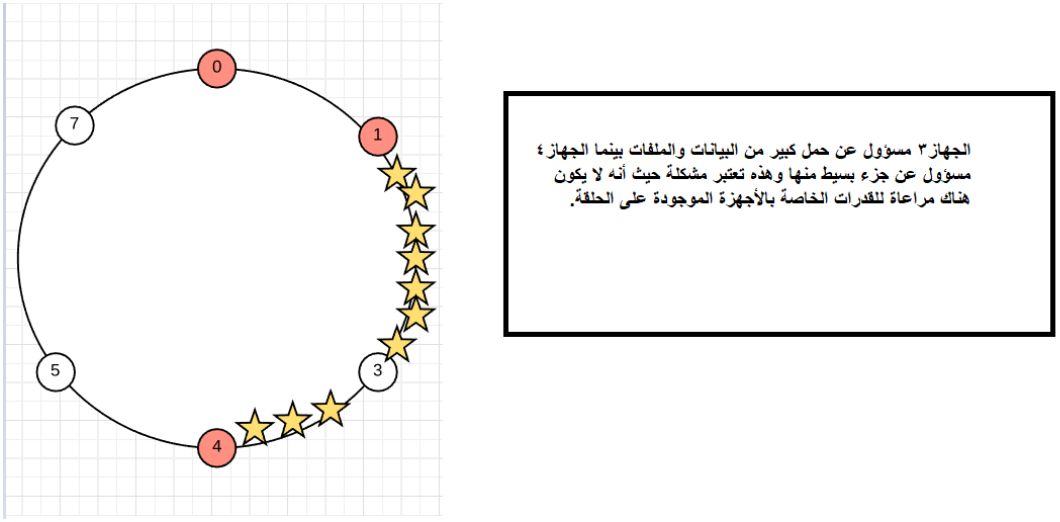
1. **قابلية التوسع (Scalability):** توفر كاساندرا قابلية توسع عالية جداً بحيث تسمح بإضافة المزيد من الأجهزة والمستخدمين وأيضا المزيد من البيانات.
2. **التخزين المرن للبيانات (Flexible data storage)**: توفر كاساندرا إمكانية تخزين مرن للبيانات بحيث تستوعب كل الأشكال المحتملة من البيانات والتي تشمل البيانات المنظمة (Structure data) ,البيانات شبه المنظمة (Semi structure data) والبيانات الغير منظمة (Unstructured data) .
3. **سهولة توزيع البيانات (Easy data distributed)**: توفر كاساندرا سهولة في عملية توزيع البيانات وذلك من خلال القيام بعمل نسخ (replicas) منها على العديد من مراكز البيانات والخوادم.

4. **الكتابة السريعة (Fast Write)**: بما أن كاساندرا مصممة لتعمل مع أجهزة بسيطة وأسعارها ليست باهظة تقوم بتوفير عملية كتابة سريعة جداً وأيضا عملية تخزين لآلاف التيرابايت من البيانات دون التأثير على كفاءة عمليات القراءة.

5.3.1 تقنيات كاساندرا (Cassandra techniques)

أولا: تقسيم البيانات (Data partitioning)

في كاساندرا توزيعة الأجهزة تكون مشابهة لتوزيعة الاجهزة في Chord DHT التي تم ذكرها سابقا في الفصل الرابع والتي تكون على شكل حلقة. فبتالي اذا لم يكن هناك تقسيم عادل للبيانات سوف يكون هناك أجهزة مسؤولة عن حمل كبير من البيانات وأجهزة تعمل على جزء بسيط منها وذلك لأنه لا يوجد مراعاة لإمكانيات الأجهزة وقدراتها كما يوضح الشكل (50).



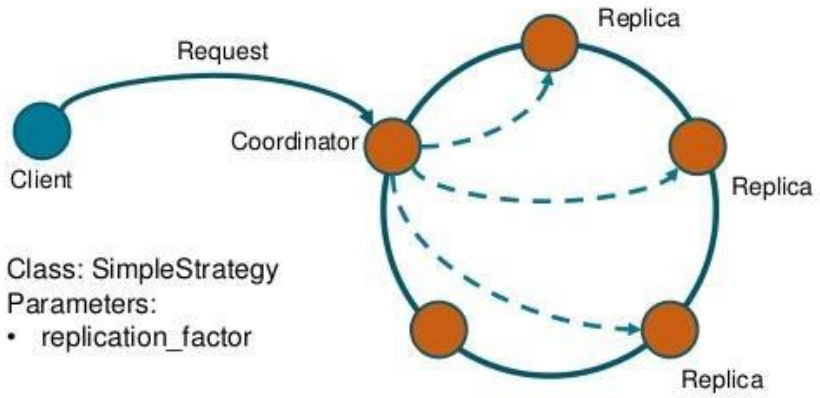
شكل 50: التوزيع غير العادل للأحمال

والحل لهذه المشكلة أن يتم توزيع البيانات على كل الأجهزة الموجودة على الحلقة بحيث يكون كل جهاز موجود مسؤول عن جزء من البيانات وتتم هذه العملية من خلال القيام بتحديد مفتاح معروف في كاساندرا Row Key ويكون هذا المفتاح مع كل جزء من البيانات حيث أنه بناءً عليه يتم تحديد الجهاز المسؤول عن جزء معين من البيانات.ويتم الحصول على عنوان الجهاز المسؤول عن البيانات من خلال ادخال هذا المفتاح إلى إقتران يسمى Partitioner والقيمة الناتجة من هذا الإقتران هي عنوان الجهاز المسؤول عن البيانات. والهدف من عملية تقسيم البيانات أن زيادة قابلية التوسع في كاساندرا من ناحية اضافة المزيد من الأجهزة والبيانات.

ثانيا: نسخ البيانات (Data replication)

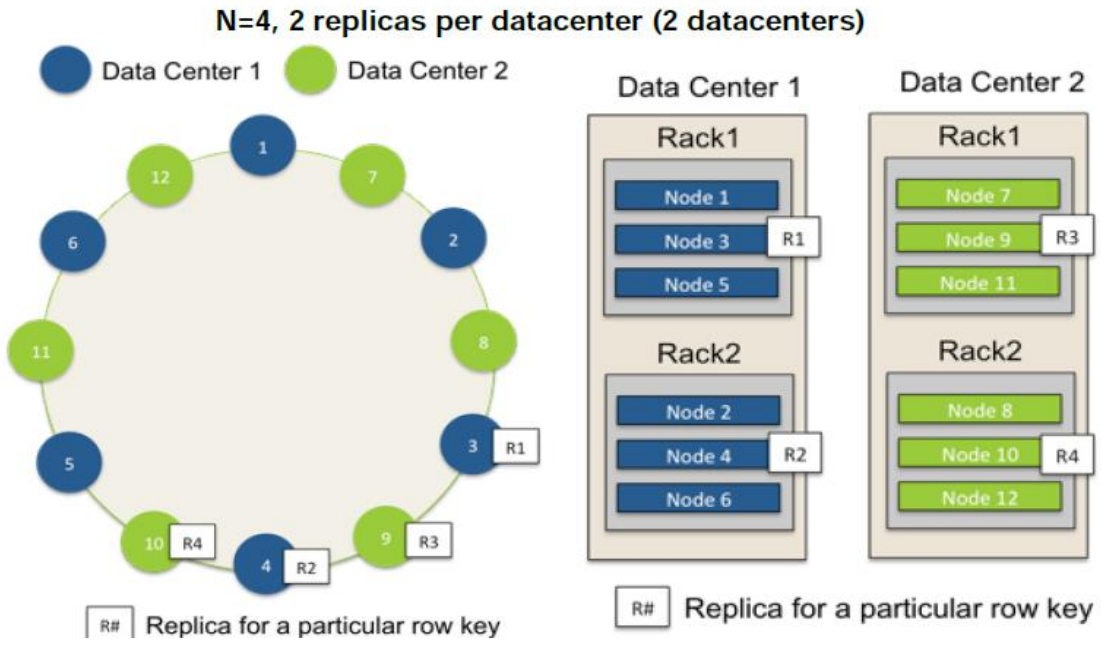
في نظام موزع مثل كاساندرا القيام بعمل نسخ للبيانات تعتبر عملية ضرورية حتى يتم زيادة توفر هذه البيانات على عدد كبير من الأجهزة ولعدد كبير من المستخدمين وأيضا حتى يزيد من متانة هذه البيانات وأن لا يتم فقدها بسهولة. يوجد اثنتين من الإستيراتيجيات لعمل نُسخ للبيانات وتتمثل بالآتي:

1. **الأستيراتيجية البسيطة (Simple strategy)**: تعتبر الإستيراتيجية الأبسط لعمل نُسخ للبيانات وتعتبر Data center and rack unaware حيث لا يكون فيها تحكم بمكان وضع هذه النُّسخ. فالنسخة الأولى تكون موجودة على الجهاز الأول الموجود على الحلقة والذي يسمى Coordinator أما النُّسخ الثانية يتم وضعها على الأجهزة المتبقية في الحلقة. فالنُّسخ التي يتم الحصول عليها هي النُّسخة الرئيسية و1-N من النُّسخ الأخرى.وتطبق هذه الإستيراتيجية عندما تكون الأجهزة التي ستقوم بتخزين النُّسخ على نفس مركز البيانات (Data center).والشكل (51) يوضح ذلك.



شكل 51: Simple strategy [5]

2. **استيراتيجية الشبكة (Network Topology strategy)**: تعتبر هذه الإستيراتيجية أكثر ذكاءً من الإستيراتيجية البسيطة حيث أنها تعتبر Data center and rack aware أي انه يوجد فيها إمكانية لتحديد مكان وضع النُّسخ. يتم تطبيق هذه الإستيراتيجية عندما تكون الأجهزة التي ستخزن النُّسخ موجودة على مراكز بيانات متعددة. وتسمح بتحديد عدد النُّسخ التي يجب أن تكون في كل مركز بيانات وتحاول توزيع البيانات بين المجمّعات (Racks) لتقليل حدوث المشاكل.في هذه الإستيراتيجية يتم وضع النُّسخة الاولى على الجهاز الرئيسي (Coordinator) وأي جهاز يكون على نفس المجمّع (Rack) سوف يتم تخطيه وتخزين النُّسخة الثانية على جهاز موجود في مجّمع آخر. الشكل (52) يوضح ذلك. حيث أنه في هذا الشكل عدد النُّسخ هو 4 وعدد مراكز البيانات هو 2 بالتالي سيكون عدد النُّسخ لكل مركز بيانات هو 2 .يحتوي كل مركز بيانات على إثنين من المجمّعات التي تحتوي على الأجهزة فبالتالي سيتم وضع نُسخة من البيانات في كل مجمّع ولا يجب وضع أكثر من نُسخة في نفس المجمّع.



شكل 52 [6]Network Topology strategy:

ثالثا: التوافقية (Consistency)

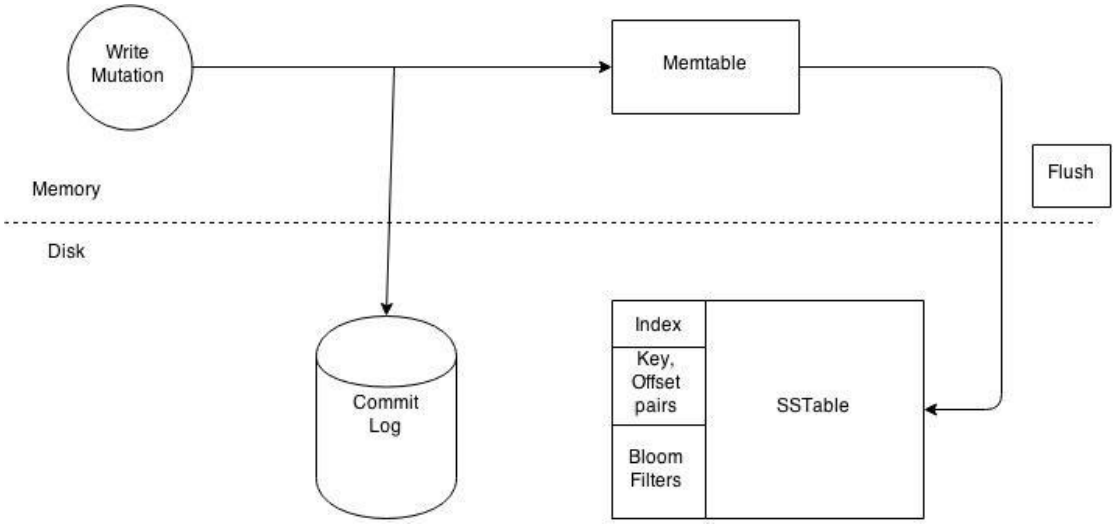
في كاساندرا تشير التوافقية الي كيفية تحديث ومزامنة البيانات الخاصة بها على كل النُّسخ الموجودة على كل الأجهزة المختلفة, ويوجد العديد من المستويات الخاصة بالتوافقية والمتمثلة بالآتي:

1. **One:** في هذا المستوى من التوافقية تكون قيمة كل من W التي تمثل عملية الكتابة و R التي تمثل عملية القراءة تساوي واحد. ويتم إعتبار أن عملية القراءة وعملية الكتابة قد نجحت إذا تمت الإستجابة من قبل جهاز واحد من الأجهزة الموجودة على الحلقة لهذه العمليات.
2. **Quorum:** تتم في هذا المستوى عمليتي القراءة والكتابة على أغلبية الأجهزة بناءً على القيمة الخاصة به والتي يتم حسابها من المعادلة التالية: (floor(n/2 +1 .
3. **All:** في هذا المستوى من التوافقية تكون قيمة كل من W و R مساوية لعدد الاجهزة الموجودة في الحلقة (W=R=N) .ففي هذه الحالة ستنجح عملتي القراءة والكتابة في حال تمت الإستجابة من قبل كل الأجهزة الموجودة علة الحلقة.

4. **Any:** في هذا المستوى من التوافقية عملية الكتابة يجب أن تتم على الاقل على جهاز واحد بحيث إذا كانت كل الأجهزة الأخرى التي تخزن نُسخ من البيانات مغلقة أن تكتمل عملية القراءة بشكل صحيح ودون مشاكل.

رابعا:التعامل مع طلبات الكتابة (Handling write request)

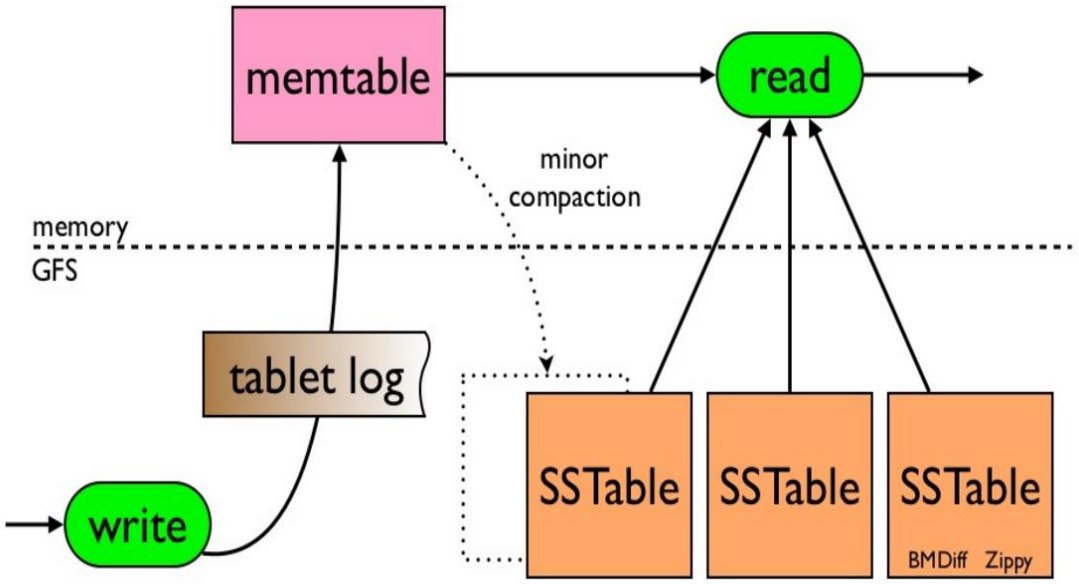
عند حدوث طلب لعملية كتابة سوف يتم إرسال هذا الطلب إلى جهاز معين بناءً على قيمة W.وعملية الكتابة التي تتم من قبل الأجهزة التي تخزن النُّسخ سيتم تسجيلها في ملف يسمى LogFile الذي يحتوي على كل العمليات التي تتم على النظام.بعد ذلك سوف يتم القيام بعملية الكتابة على الذاكرة الرئيسية الخاصة بالجهاز الذي يقوم بعملية التخزين والكتابة.بعد انتهاء عملية الكتابة تكون نجحت هذه العملية بشكل محلي. ومن فترة إلى أخرى سيتم إرسال البيانات القديمة من الذاكرة الرئيسية إلى الذاكرة الثانوية وذلك لأن الطلب عليها أصبح قليل والشكل (53) يوضح هذه العملية.



شكل 53 :Handling write request[7]

خامسا: التعامل مع طلبات القراءة (Handling read request)

البيانات تكون مخزنة في جداول تسمى SStables على شكل أعمدة ويكون لهذه البيانات مفتاح خاص يدل عليها يسمى Row Key. البيانات التي تكون خاصة بمفتاح معين يمكن أن تكون مخزنة على عدد من جداول SStables بالتالي حتى تتم عملية القراءة يجب أن يتم قراءة كل البيانات التي لها نفس المفتاح من كل جداول SStables وبعد ذلك تجميع البيانات وإرسالها الي Coordinator.والشكل (54) يوضح هذه العملية.



شكل 54:Handling read request[8]

5.5 تجربة كاساندرا

5.5.1 وصف التجربة

صممت هذه التجربة حتى يتم توضيح المفاهيم الأساسية لقاعدة البيانات الغير علائقية التي تسمى كاسندرا.تمت إتمام هذه التجربة على جهازين (2 nodes) على نظام التشغيل أوبنتو (Ubuntu), تم تسمية الأجهزة كالآتي Cassandra1 و Cassandra2. في هذه التجربة سوف يتم انشاء قاعدة بيانات على إحدى هذه الأجهزة وعمل نسخ لها على الجهاز الآخر من خلال عملية النَّسخ المعروفة في كاساندرا, وستم تحديد عدد النُّسخ بناءً على خاصية تسمى Replication Factor وبما أنه يوجد جهازين في هذه التجربة فإن عدد النُّسخ هو 2. بعد التأكد من وجود النُّسخ على كلا الجهازين سوف يتم توضيح عمليات القراءة والكتابة كيف تتم عليها من خلال التحكم بالمتغيرات التالية : المتغير W الذي يمثل عملية الكتابة و المتغير R الذي يمثل عملية القراءة. وخلال هذه التجربة قمنا بالحالات الآتية:

1. W=2: في هذه الحالة سوف تتم عملية الكتابة بنجاح و دون أي عوائق لأن عدد الأجهزة الموجودة هو 2
2. W=3: في هذه الحالة عملية الكتابة لن تتم بالشكل المطلوب وذلك لأن عدد الأجهزة الموجودة هو 2 بالتالي عند عدم قدوم الإستجابة المطلوبة لهذا الطلب بناءً على قيمة W سوف يحدث مشكلة ولن تتم عملية الكتابة بالشكل المطلوب.
3. R=2: في هذه الحالة ستتم عملية القراءة وإسترجاع البيانات بنجاح ودون مشاكل وذلك لأن عدد الأجهزة الموجود مطابق لقيمة R.
4. أثناء عملية القراءة تم إغلاق كاسندرا على الجهاز الثاني وقمنا بعملية القراءة من جديد وكانت قيمة R=2. في هذه الحالة لن تنجح عملية القراءة لأن عدد الاجهزة الموجودة هو 1.ولحل هذه المشكلة والقيام بعملية القراءة بالشكل الطبيعي نجعل قيمة R=1.

5.5.2 الهدف من التجربة

الهدف الأساسي من هذه التجربة هو توضيع العمليات التي تتم في كاساندرا للطلاب بشكل أفضل, وتوضيح مستويات التوافقية فيها وأيضا توضيح مفهوم النَّسخ على الأجهزة الأخرى.

5.5.3 المتطلبات اللازمة لتطبيق التجربة

تتمثل متطلبات التجربة بالآتي:

1. نظام تشغيل ابنتو (Linux Ubuntu 16.04), في التجربة تم الاعتماد على vmware لاحتضان الابنتو. وذلك لأن vmware يوجد بها اتصال بين أجهزة الأبنتو بداخله .
2. جافا (java jdk) بأي إصدار, وفي هذه التجربة تم استخدام الإصدار الثامن.
3. بايثون بأي إصدار . في التجربة تم الاعتماد على الإصدار الثاني.
4. كاسندرا الإصدار 3.11.2 .

5.5.4 تنفيذ التجربة

حتى يتمكن الطالب من تنفيذ هذه التجربة يجب أن يقوم أولا بتنصيب وإعداد كاسندرا من خلال مجموعه من الأوامر التي سوف يتم ذكرها لاحقاً.

5.5.5 أوامر التجربة

هنا سيتم ذكر الأوامر التي تم تطبيقها في هذه التجربة:

سوف يكون لدينا جهازان الأول cassandra1 ذو العنوان 192.168.144.138 الثاني cassandra2 ذو العنوان 192.168.144.139	
أولاً	
التحديث للنظام والتطبيقات الخاصة بالابنتو من خلال تحديث لملف source list.	$ sudo apt-get update
تنزيل الإصدار الافتراضي للجافا غالبا يكون اخر اصدار عن طريق repository أو المستودع الخاص بالابنتو .	$ sudo apt-get install default-jdk
ملاحظة : في حال فشل تنزيل الجافا, يجب إعادة تشغيل النظام والمحاولة مره أخرى.	
لفحص إصدار الجافا	$ java -version
تنزيل البايثون, وذلك من أجل أداة csqsh سوف يتم ذكرها لاحقاً .	$ sudo apt-get install -y python3-pip
لفحص إصداء البايثون .	$ python -version

تنزل sshاختصار ل Secure Shell وذلك من أجل اتصال الأجهزة عن بعد.	$ sudo apt-get install ssh
	$ ssh-keygen -t rsa -P ''
	$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

للفحص أنه يعمل	$ sshlocalhost
ثانياً	
نقوم بتنصيب كاسندرا إصدار 3.11.2 من الموقع الخاص بهم, من خلال الرابط : http://www.apache.org/dyn/closer.lua/cassandra/3.11.2/apache-cassandra-3.11.2-bin.tar.gz نذهب الى التنزيلات ونفك ضغط الملف التنصيب	

ثالثاً	
نعدل على الملف cassandra.yml ذو الامتدات /downloads/apache-cassandra-3.11.2/conf/ cassandra.yml	
seeds: "192.168.144.138,192.168.144.139"	seeds :تضم قائمة عناوين الأجهزة لاستخدامها كنقاط اتصال عند الانضمام الى الكلستر .
listen_address: 192.168.144.138	listen_address :هو عنوان الجهاز الذي يستخدمة الجهاز الاخر(node) للاتصال مع هذا الجهاز .
rpc_address: 192.168.144.138	rpc_address :هو عنوان الاستماع من اجل الاستدعاءات عن بعد .
endpoint_snitch: RackInferringSnitch	endpoint_snitch :المجموعات التي تستخدمها كاسندرا من أجل تحديد مواقع الأجهزة .
رابعاً	
$ sudoiptables -A INPUT -p tcp -s 192.168.144.138 -m multiport --dports 7000,9042 -m state --state NEW,ESTABLISHED -j ACCEPT $ sudoiptables –L	Iptables :هو جدار حماية ، مثبت بشكل افتراضي على جميع توزيعات Ubuntu . نستخدمة هنا من أجل تدفق البيانات عبر الأجهزة (nodes) من خلال تفعيل المنافذ (port) 7000,9042.
إعدادات الجهاز الأخر	
نكمل إعدادات الجهاز الأخر بنفس الطريقة .لكن الاختلاف يكون بكل من : listen_address: 192.168.144.139 rpc_address: 192.168.144.139	
خامساً	
$ ./cassandra -f	لتشغيل كاسندرا من داخل ملف ذو الامتداد /downloads/apache-cassandra-3.11.2/bin
$ ./nodetool -p 7199 status	لفحص الأجهزة المتصلة من داخل المسار /downloads/ apache-cassandra-3.11.2/bin
بعد تشغيل كساندرا على كل من الجهازان, نستخدم أداة cqlsh التي تثبت مع كساندرا . يتم استخدامها من أجل انشاء الداتا بيز والجداول لإدخال البيانات عليها .	
$ ./cqlsh 192.162.144.138	للانتقال إلى داخل الأداة ننتقل على المسار /downloads/ apache-cassandra-3.11.2/bin ونضع عنوان الجهاز.

سادساً	
cqlsh> CREATE KEYSPACE ppu WITH REPLICATION = {'class' :'SimpleStrategy', 'replication_factor': 2};	لإنشاء داتا بيز لتكون ppu . النسخ عددها 2 بناءاً على عدد الأجهزة.
cqlsh> use ppu;	لنستخدم الداتا بيز .
cqlsh:ppu> CREATE TABLE students(id int PRIMARY KEY, name varchar, major varchar, avgint);	لإنشاء جدول ليكن student .
cqlsh:ppu> INSERT INTO students (id, name, major,avg) VALUES (141028,'ShYma','IT',90);	لإضافة قيم على الجدول .
cqlsh:ppu> SELECT * FROM students;	لاسترجاع القيم .
cqlsh:ppu> consistency (number);	للتغير على كل من w,r يتم من خلال الأمر . ونضع داخل number عدد كل من w,r

جدول 6: أوامر تجربة كاسندرا

5.5.6 نتائج تنفيذ التجربة

فيما يلي توضيح للنتائج التي ظهرت عند تنفيذ الاوامر الخاصة بالتجربة:

1. الشكل (55) يوضح تشغيل كاساندرا.

```
cloud@cloud-virtual-machine:~/Downloads/apache-cassandra-3.11.2/bin$ ./cassandra -f

CompilerOracle: dontinline org/apache/cassandra/db/Columns$Serializer.deserializeLargeSubset (Lorg/apache/
cassandra/io/util/DataInputPlus;Lorg/apache/cassandra/db/Columns;I)Lorg/apache/cassandra/db/Columns;
CompilerOracle: dontinline org/apache/cassandra/db/Columns$Serializer.serializeLargeSubset (Ljava/util/Col
lection;ILorg/apache/cassandra/db/Columns;ILorg/apache/cassandra/io/util/DataOutputPlus;)V
CompilerOracle: dontinline org/apache/cassandra/db/Columns$Serializer.serializeLargeSubsetSize (Ljava/util
/Collection;ILorg/apache/cassandra/db/Columns;I)I
CompilerOracle: dontinline org/apache/cassandra/db/commitlog/AbstractCommitLogSegmentManager.advanceAlloca
tingFrom (Lorg/apache/cassandra/db/commitlog/CommitLogSegment;)V
CompilerOracle: dontinline org/apache/cassandra/db/transform/BaseIterator.tryGetMoreContents ()Z
CompilerOracle: dontinline org/apache/cassandra/db/transform/StoppingTransformation.stop ()V
CompilerOracle: dontinline org/apache/cassandra/db/transform/StoppingTransformation.stopInPartition ()V
CompilerOracle: dontinline org/apache/cassandra/io/util/BufferedDataOutputStreamPlus.doFlush (I)V
CompilerOracle: dontinline org/apache/cassandra/io/util/BufferedDataOutputStreamPlus.writeExcessSlow ()V
CompilerOracle: dontinline org/apache/cassandra/io/util/BufferedDataOutputStreamPlus.writeSlow (JI)V
CompilerOracle: dontinline org/apache/cassandra/io/util/RebufferingInputStream.readPrimitiveSlowly (I)J
CompilerOracle: inline org/apache/cassandra/db/rows/UnfilteredSerializer.serializeRowBody (Lorg/apache/cas
sandra/db/rows/Row;ILorg/apache/cassandra/db/SerializationHeader;Lorg/apache/cassandra/io/util/DataOutputP
lus;)V
```

شكل 55:تشغيل كاساندرا

2. الشكل (56) يوضح الأجهزة المتصلة في التجمع (cluster) . يوجد لدينا جهازان up

```
cloud@cloud-virtual-machine:~/Downloads/apache-cassandra-3.11.2/bin$ ./nodetool -p 7199 status
Datacenter: 168
===============
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address          Load       Tokens       Owns    Host ID                               Rack
UN  192.168.144.138  313.99 KiB  256          ?       2db81844-5fdc-4dce-afcb-719a31b2a280  144
UN  192.168.144.139  172.59 KiB  256          ?       07e9b590-ff27-4850-9d00-c96ceca22342  144
```

شكل 56:الأجهزة المتصلة بالمُجمع (cluster)

3. يوضح الشكل (57) الوصول إلى أداة cqlsh, وإنشاء داتا بيز واستخدامها.

```
cloud@cloud-virtual-machine:~/Downloads/apache-cassandra-3.11.2/bin$ ./cqlsh 192.168.144.138
Connected to Test Cluster at 192.168.144.138:9042.
[cqlsh 5.0.1 | Cassandra 3.11.2 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh> CREATE KEYSPACE ppu WITH REPLICATION = {'class' :'SimpleStrategy', 'replication_factor': 2};
AlreadyExists: Keyspace 'ppu' already exists
cqlsh> CREATE KEYSPACE ppu1 WITH REPLICATION = {'class' :'SimpleStrategy', 'replication_factor': 2};
cqlsh> use ppu1;
```

شكل :57الدخول إلى cqlsh وإنشاء داتا بيز

4. يوضح الشكل (58) إنشاء جدول.

```
cqlsh:ppu1> CREATE TABLE students(
       ... id int PRIMARY KEY,
       ... name varchar,
       ... major varchar,
       ... avg int);
```

شكل 58:إنشاء جدول Student

5. يوضح الشكل (59) لإضافة قيم واسترجاعها.

```
cqlsh:ppu1> INSERT INTO students (id, name, major,avg) VALUES (141028,'ShYma','IT',90);
cqlsh:ppu1> SELECT * FROM students;

 id     | avg | major | name
--------+-----+-------+-------
 141028 |  90 |    IT | ShYma

(1 rows)
```

شكل 59: إضافة قيم في الجدول وإسترجاعها

6. لتكن عملية الكتابة على ثلاثة أجهزة ,(W=3) ويوجد فقط لدينا جهازان . بالتالي لن تنجح عملية الكتابة, ويتضح ذلك من خلال الشكل (60) .

```
cqlsh:ppu1>  consistency three;
Consistency level set to THREE.
cqlsh:ppu1> INSERT INTO students (id, name, major,avg) VALUES (141040,'Ali','CS',80);
NoHostAvailable:
```

شكل 60:عملية الكتابة على 3 أجهزة

7. لنغير مستوى التوافقية (consistency) ولتكن اثنان (W=2), ونقم بإضافة طالب جديد , بما انه يوجد لدينا جهازان سوف يتم الاضافه بلا مشاكل يتضح من خلال الشكل (61).

```
cqlsh:ppu1> consistency two;
Consistency level set to TWO.
cqlsh:ppu1> INSERT INTO students (id, name, major,avg) VALUES (141040,'Ali','CS',80);
cqlsh:ppu1> SELECT * FROM students;

 id     | avg | major | name
--------+-----+-------+-------
 141040 |  80 |    CS |   Ali
 141028 |  90 |    IT | ShYma

(2 rows)
```

شكل 61:عملية الكتابة على جهازين

8. قمنا بإيقاف cassandra2 ونفحص الأجهزة وكما هو موضح بالشكل (62) فإن الجهاز الثاني down.

```
cloud@cloud-virtual-machine:~/Downloads/apache-cassandra-3.11.2/bin$ ./nodetool -p 7199 status
Datacenter: 168
===============
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address          Load       Tokens       Owns    Host ID                               Rack
UN  192.168.144.138  372.61 KiB  256          ?       2db81844-5fdc-4dce-afcb-719a31b2a280  144
DN  192.168.144.139  231.21 KiB  256          ?       07e9b590-ff27-4850-9d00-c96ceca22342  144
```

شكل 62:لحظة ايقاف الجهاز الثاني

9. وعندما استرجاع البيانات تكون R=2 وأحد الأجهزة معطل يرفض عملية القراءة يتضح من خلال الشكل (63).

```
cqlsh:ppu1> consistency ;
Current consistency level is TWO.
cqlsh:ppu1> SELECT * FROM students;
NoHostAvailable:
```

شكل 63:استرجاع البيانات وأحد الأجهزة معطل

10. وعند تغير R=1 يمكنه القراءة من جهاز واحد وهو cassandra1.

```
cqlsh:ppu1> consistency one;
Consistency level set to ONE.
cqlsh:ppu1> SELECT * FROM students;

 id     | avg | major | name
--------+-----+-------+-------
 141040 |  80 |    CS |   Ali
 141028 |  90 |    IT | ShYma

(2 rows)
```

شكل 64:القراءة من جهاز واحد

5.5.7 ملحقات التجربة

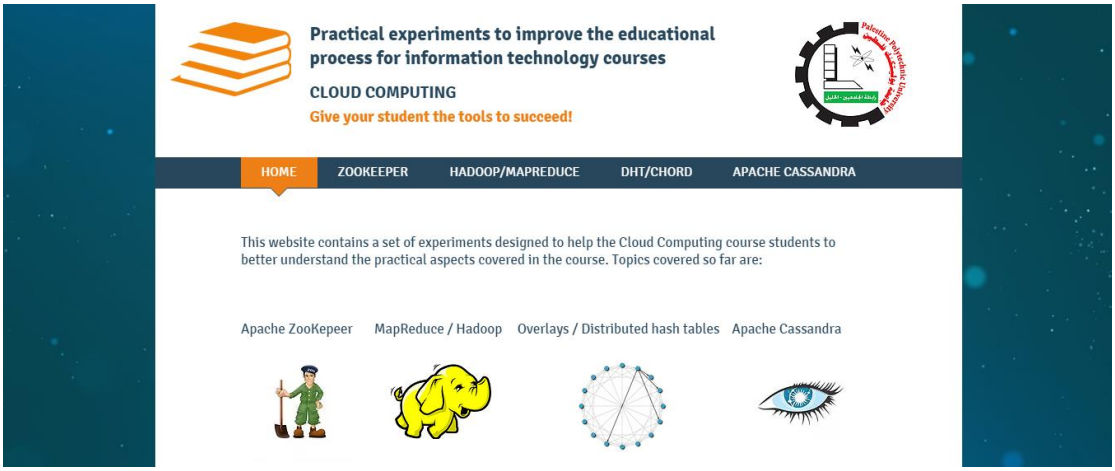
لقد تم تزويد هذه التجربة بالعديد من الملحقات التي تعتبر مساعِدة للطالب حتى يتمكن من فهم هذه التجربة, وتمثلت هذه الملحقات بالآتي:

1. فيديو خاص بإعدادات وتنفيذ التجربة (Execution video)
https://www.youtube.com/watch?v=HuKg4xf3YpI&t=785s

الفصل السادس: واجهات الموقع الالكتروني

يتضمن هذا القسم واجهات الموقع الالكتروني الذي يحتوي التجارب.

وتم بناء الواجهات من خلال البرنامج Wix.



Practical experiments to improve the educational process for information technology courses

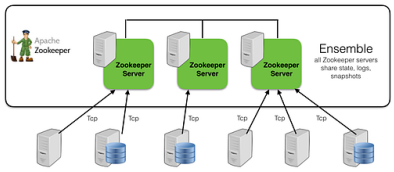
CLOUD COMPUTING

Give your student the tools to succeed!

HOME | ZOOKEEPER | HADOOP/MAPREDUCE | DHT/CHORD | APACHE CASSANDRA

Apache ZooKeeper is an open source file application program interface (API) that allows distributed processes in large systems to synchronize with each other so that all clients making requests receive consistent data . Zookeeper uses a distributed consensus protocol to determine which node in the ZooKeeper service is the leader at any given time.

The leader assigns a timestamp to each update to keep order. Once a majority of nodes have acknowledged receipt of a time-stamped update, the leader can declare a quorum, which means that any data contained in the update can be coordinated with elements of the data store. The use of a

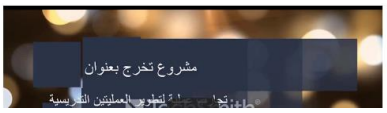


Source : ZooKeeper definition

Experiment

The idea of the experiment is to perform a simple summation of a range of numbers in a distributed way. For more details, please watch this illustrative video.

To see how the experiment works, please watch this demo video.



To see how the experiment works, please watch this demo video.

The corresponding programming codes and configuration files can be found here.

Docker-Zookeeper

@ 2018 by HALA TALAHMEH & NOOR AMRO & SHAYMA NATSHE

Practical experiments to improve the educational process for information technology courses

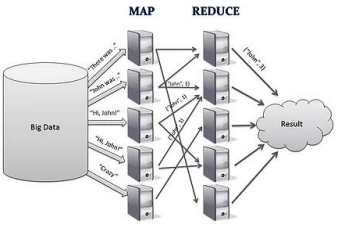
CLOUD COMPUTING

Give your student the tools to succeed!

HOME | ZOOKEEPER | HADOOP/MAPREDUCE | DHT/CHORD | APACHE CASSANDRA

Hadoop is a collection of open-source software utilities that facilitate using a network of many computers to solve problems involving huge amounts of data and it provides a software framework for distributed storage and processing of big data using the MapReduce .

At the heart of Hadoop is MapReduce, the programmatic framework that makes it easy to write applications that process vast amounts of data, parallel to large sets of devices (nods).MapReduce is one of two main components of Hadoop. These are HDFS and YARN.



Source : MapReduce_Architecture

Experiment

Word Count Experiment Counts the number of times each word appears in a particular input group (often a large amount of Documents), in our experience a book was chosen to calculate the number of impressions each word has. For more details, please watch this illustrative video.

To see how the experiment works, please watch this demo video.

The corresponding programming codes and configuration files can be found here.

WordCount-MapReduce

Practical experiments to improve the educational process for information technology courses

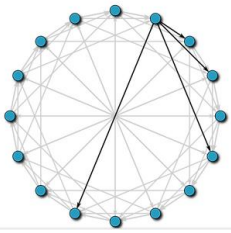
CLOUD COMPUTING

Give your student the tools to succeed!

HOME | ZOOKEEPER | HADOOP/MAPREDUCE | DHT/CHORD | APACHE CASSANDRA

Chord is a protocol and algorithm for a peer-to-peer distributed hash table. A distributed hash table stores key-value pairs by assigning keys to different computers (known as "nodes"); a node will store the values for all the keys for which it is responsible.

Chord specifies how keys are assigned to nodes, and how a node can discover the value for a given key by first locating the node responsible for that key. Chord solve a general and fundamental problem in the peer to peer network which is locating a data item in a collection of distributed nodes, considering frequent node arrivals and departures.



Source : Chord definition

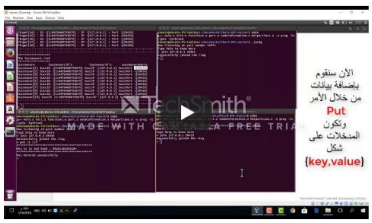
Source : Chord_Architecture

Experiment

The idea of this experiment is to perform the primary operation of the Chord distributed hash table Which is represented in Creating the Chord ring , Joining nodes to this ring ,Put data in the nodes and Get this data. For more details, please watch this illustrative video

To see how the experiment works, please watch this demo video.

To see how the experiment works, please watch this demo video.



The corresponding programming codes and configuration files can be found here.

Chord-Distributed-Hash-Table

Practical experiments to improve the educational process for information technology courses

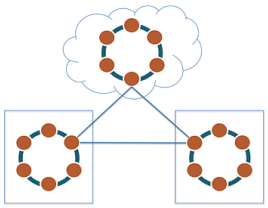
CLOUD COMPUTING

Give your student the tools to succeed!

HOME | ZOOKEEPER | HADOOP/MAPREDUCE | DHT/CHORD | APACHE CASSANDRA

Apache Cassandra is a free and open-source distributed NoSQL database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure.

Cassandra offers robust support for clusters spanning multiple datacenters, with asynchronous masterless replication allowing low latency operations for all clients.

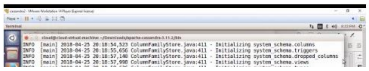


Source : Cassandra definition

Experiment

The main goal of this experiment is to provide a better description of how cassandra operation are performed,then it's describe the consistancy levels and the replication conceptin cassandra on more than one node . For more details, please watch this illustrative video

To see how the experiment works, please watch this demo video.



To see how the experiment works, please watch this demo video.

The corresponding programming codes and configuration files can be found here.

Apache_Cassandra

الفصل السابع: الإستنتاجات (Conclusions)

7.1 النتائج النهائية

في نهاية هذا المشروع, نحن سعداء بقولنا ان المشروع قد حقق جزء كبير من أهدافه. حيث أننا الآن نمتلك مرجع أساسي لطلبة مساق الحوسبة السحابية المتمثل في موقع ويب يحتوي على مجموعة من التجارب العملية والفيديوهات التوضيحية والتنفيذية لهذه التجارب والتي تهدف إلى توضيح التقنيات المطروحة في هذا المساق والمتمثلة بالآتي:

1. **Apache Zookeeper**
2. **Distributed hash tables(DHT) / Chord DHT**
3. **Apache Hadoop**
4. **Apache Cassandra**

ويجدر بالذكر هنا أن بعض التجارب تم إعطاءها لطلاب المساق في الفصل الماضي لأداءها وتجربتها بأنفسهم. ونجحت تقريباً بنسبة 90% وتم تعويض 10% المتبقية لأخذ الملاحظات الازمة للتحسين على التجارب.

وأيضاً فخورين بهذا الإنجاز الأول من نوعه على مستوى جامعة بوليتكنك فلسطين.

7.2 العمل المستقبلي

1. القيام بعمل ملف تنفيذي (Bash file) خاص بكل التجارب .
2. توضيح تقنيات Apache Hive, Apache PIG, Spark, Virtualization .
3. دعم المفاهيم العلمية النظرية بالمزيد من التجارب العملية .
4. القيام بالتجارب الحالية بعدد أكبر من الأجهزة (nodes) .

المصادر

الفصل الثاني

[1] https://www.youtube.com/watch?v=31jzI4b8PiY

[2] https://www.spantree.net/blog/2015/04/29/10-things-to-know-about-docker.html

- https://www.tutorialspoint.com/zookeeper/index.htm
- ZooKeeper: Wait-free coordination for Internetscale systems by Hunt et al., 2010

الفصل الثالث

[3] Cloud Computing course (15-319) slides, CM Qatar.

[4] https://www.guru99.com/introduction-to-mapreduce.html

- https://www.tutorialspoint.com/hadoop/index.htm
- https://www.tutorialspoint.com/map_reduce/index.htm
- http://chaalpritam.blogspot.com/2015/01/
- https://www.researchgate.net/profile/Abdelrahman_Osman3/publication/318866503_Hadoop_for_Beginners_-_in_Arabic/links/59906a48a6fdcc10d8114874/Hadoop-for-Beginners-in-Arabic.pdf
-

الفصل الرابع

- https://en.wikipedia.org/wiki/Chord_(peer-to-peer)
- http://nms.csail.mit.edu/papers/chord.pdf
- https://en.wikipedia.org/wiki/Distributed_hash_table

الفصل الخامس

[5] https://www.slideshare.net/DataStax/replication-and-consistency-in-cassandra-what-does-it-all-mean-christopher-bradford-datastax-c-summit-2016

[6] Slides of Distributed Systems and Cloud Computing Course, Eurecom (by Marko Vukolić)

[7] https://dzone.com/articles/introduction-apache-cassandras

[8] https://www.slideshare.net/quipo/nosql-databases-why-what-and-when/124

- https://www.tutorialspoint.com/cassandra/cassandra_introduction.htm
- https://searchstorage.techtarget.com/definition/cloud-storage
- https://www.linkedin.com/pulse/introduction-apache-cassandras-architecture-abdul-basith

ملاحظة : تم الوصول إلى جميع الروابط أعلاه بتاريخ 2018\4\29