Introduction to IBM Cloudant- DBaaS

Week-4 **Part 1: Basics of Cloudant || Part 2: Working with Cloudant**

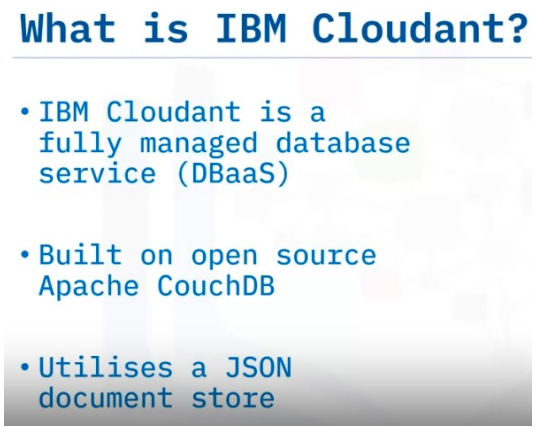
Database-as-a-service (DBaaS) is a popular solution for hybrid multicloud applications. IBM Cloudant is a fully managed DBaaS built on open source Apache CouchDB. Cloudant aims to be the data layer for all your web and mobile applications. In this module, you will find out how simple developing modern web applications is with Cloudant's rich features and JSON document store.You will explore the architecture of Cloudant as a NoSQL database. You will gain hands-on experience with Cloudant capabilities and key technologies. And you will learn how to use the Cloudant dashboard to create and manage your database.

Learning Objectives :

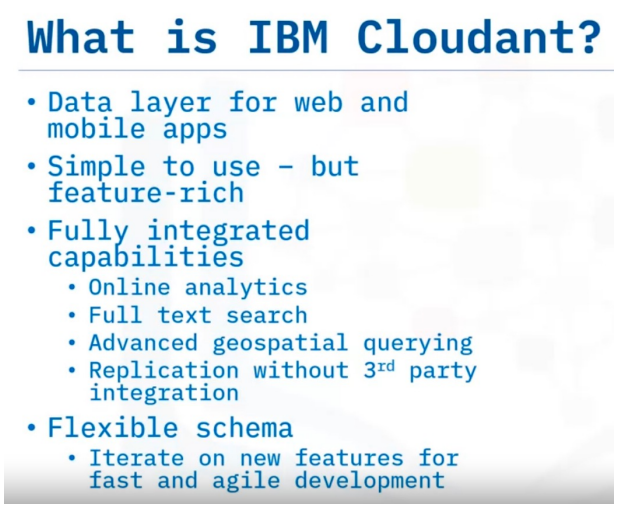
- Describe IBM Cloudant, its key characteristics, and its key capabilities.
- Describe the IBM Cloudant architecture and technologies.
- Identify the key benefits of IBM Cloudant and its common use cases.
- Detail the different deployment plans for IBM Cloudant.
- Perform basic operations on the IBM Cloudant dashboard.
- Create an IBM Cloud account.
- Create and work with a Cloudant database.
- Describe the elements that users can manage in the Cloudant dashboard.
- Replicate databases and monitor tasks and instances in the Cloudant dashboard.
- Describe the characteristics of documents in a Cloudant database.
- Insert, query, and modify database documents in the Cloudant dashboard.
- Describe the HTTP API in IBM Cloudant.
- Explain what curl is.
- Discuss HTTP methods used in curl.
- Summarize when and how to use text and JSON indexes in IBM Cloudant.
- Use HTTP API to create and query Cloudant databases.

Overview of Cloudant

- ❑ Describe what IBM Cloudant is and its key characteristics
- ❑ Describe Cloudant's replication capabilities List key capabilities provided by Cloudant
- ❑ Describe how to access IBM Cloudant



- IBM Cloudant is a fully managed database service, or database-as-a-service (DBaaS), for hybrid multicloud applications.
- It is built on open source Apache CouchDB and offers a fully compatible HTTP API without the need for proprietary drivers.
- It also utilises a JSON document store, so development is simplified for modern Web applications by dealing with JSON objects on all layers of the stack



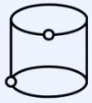
- Cloudant aims to be the data layer for all your Web and mobile applications, and it is simple to use, and yet still, rich in features.
- Cloudant has fully-integrated capabilities for online analytics, full text search, advanced geospatial querying, and replication without the need for 3rd-party integrations.
- Cloudant uses a flexible schema which means you can iterate on new features for fast and agile development.

What is IBM Cloudant?

- Distributed database optimized for large workloads
  - Web, mobile, IoT, serverless apps
- SLA-backed DBaaS means no admin overhead
  - White glove service from big data experts
- Cloudant database is securely hosted in the cloud
  - Globally controlled 24/7 by big data specialists
- Engineers handle day-to-day monitoring and maintenance
  - You focus on building apps

- Cloudant is a **distributed database** and is optimized for processing the substantial workloads that are characteristic of sizable, fast-growing web, mobile, IoT, and serverless applications.
- Because it is offered as a **Service Level Agreement-backed Database-as-a-Service**, there are no administrative overheads.
- Cloudant offers **white glove service** from big data experts to help you build your applications.
- Your Cloudant database is securely hosted in the cloud and globally controlled by big data specialists on a 24/7 basis.
- Engineers at IBM handle all the day-to-day database monitoring and infrastructure maintenance, so you can focus on building your applications.

Cloudant Capabilities

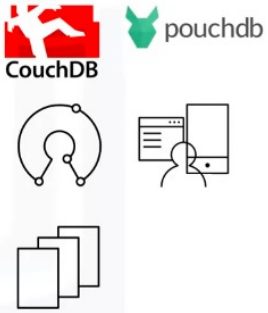


- Data replication technology is engineered to reduce database scalability risks, costs, and distractions
- Enables developers to focus on needs of business apps

- Cloudant data **replication technology** provides options engineered to **reduce scalability risks, costs, and distractions.**
- These capabilities enable developers to focus on the business's applications.

Cloudant Capabilities

- Powerful replication protocol and API compatible with open source ecosystems
- Also compatible with open source libraries for most prevalent mobile and web development stacks
- Cloudant and CouchDB sharing a common replication protocol allows developers to sync copies of Cloudant data to remote CouchDB instances at push of a button



- The powerful replication protocol and API are compatible with open source ecosystems such as Apache CouchDB and PouchDB and are also compatible with the open source libraries for the most prevalent mobile and web development stacks.
- As Cloudant and CouchDB share a common replication protocol, this allows developers to synchronize copies of their Cloudant data to a remote CouchDB instance at the push of a button.

Cloudant Capabilities

- **Offline First** mobile web app capabilities, built with **Cloudant Sync**, enable mobile data synchronization
- Android and iOS users can work offline and mobile, using stored local data, and then synchronize their data to the cloud databases

- Cloudant Search **implements** Apache Lucene for search speed and simplicity, and **Cloudant Geospatial GeoJSON storage** supports the encoded geographic data structures for built-in spatial querying and map visualization tasks.
- Offline First capabilities, built with Cloudant Sync enable mobile data synchronization, which enables application users to work offline and mobile, using stored local data, and then synchronize their data to the cloud databases.

Cloudant Capabilities

- Develop your own applications using **language-specific libraries** (wrappers that help you work with an API)

- You can develop your **own applications** using language-specific libraries (wrappers that help you work with an API).

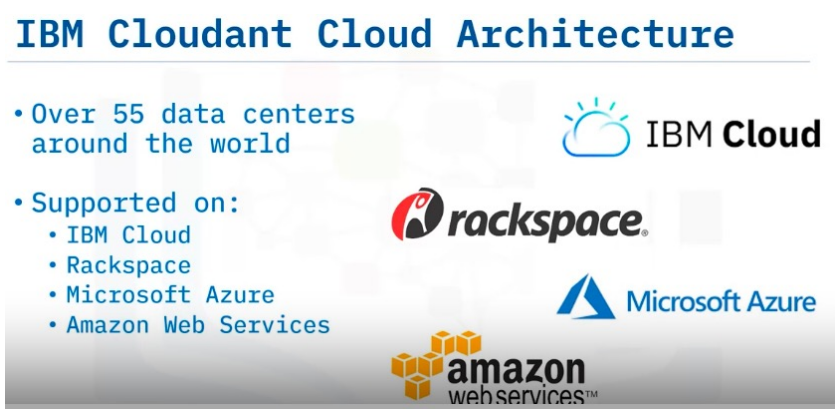
Summary

In this video, you learned that:

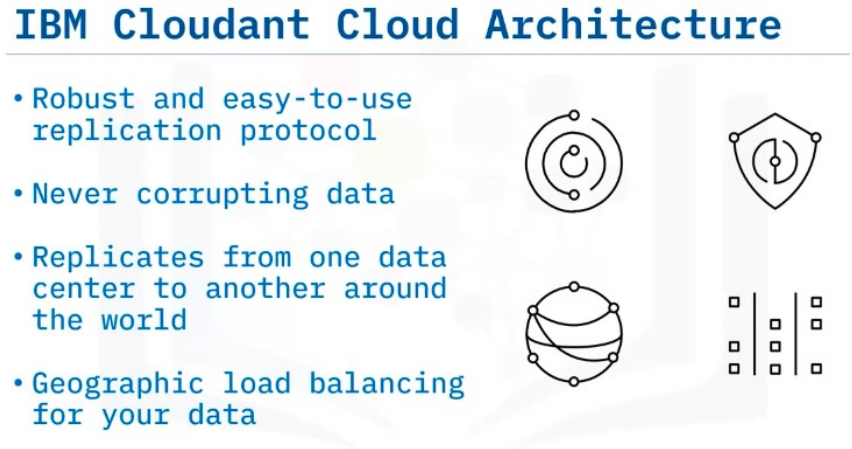
- IBM Cloudant is a fully managed DBaaS built on Apache CouchDB that uses a JSON document store
- Cloudant is a distributed database optimized for large workloads; web, mobile, IoT, serverless apps
- SLA-backed DBaaS means no admin overhead and Cloudant databases are securely cloud-hosted
- Cloudant offers powerful replication protocol and API compatible with open source ecosystems and libraries
- Cloudant provides search, geospatial querying, offline and mobile, and language-specific library capabilities

Cloudant Architecture and Key Technologies

- ❑ Describe the cloud architecture behind IBM Cloudant
- ❑ Describe the key technology components that IBM Cloudant provides



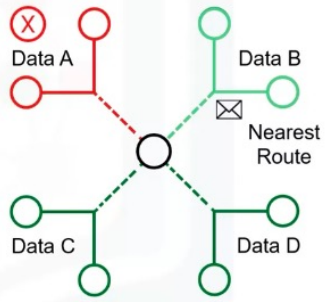
- You can host your data in more than 55 different data centers distributed all over the world – so IBM Cloudant offers a global data presence.
- IBM Cloudant is supported on: IBM Cloud (an IBM Infrastructure as a Service offering), RackSpace, Microsoft Azure, and Amazon Web Services - so it's cloud agnostic .



- Cloudant has the most robust and easy-to-use replication protocol; never corrupting the data, thereby providing the best of any available data layer.
- Cloudant replicates from one data center to another making the data available throughout the world, thus providing geographic load balancing for your data.
- All Cloudant instances are deployed on clusters that span regional availability zones, where supported, for added durability at no further charge to an organization.

IBM Cloudant Cloud Architecture

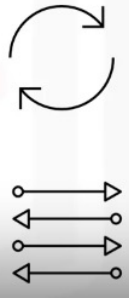
- If a data center goes offline - requests get routed to another active data center
  - High availability
  - Disaster recovery
  - Optimal performance
- Users are routed to 'closest' data center



- If one data center goes offline, the requests will get routed to another active data center, providing **high availability and disaster recovery**, as well as **optimal performance.**
- **Users get routed** to the **data center 'closest' to them** – or more accurately – by using ping timing, users actually get routed to the data center 'fastest' to them, i.e. closest in terms of time not distance.

IBM Cloudant Cloud Architecture

- Many web and mobile applications require offline sync
- Cloudant's replication protocol is compatible with:
  - Common libraries in your browser
  - Cloudant-provided libraries

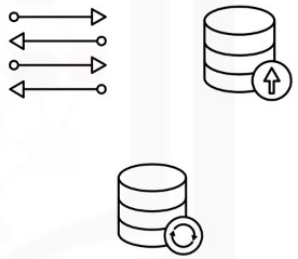


- For web and mobile applications that **require offline sync** (such as completing an expense report, or filling out a sales order form offline), Cloudant uses a **replication protocol** that is compatible with common libraries that exist in your browser, or with Cloudant-provided libraries.

IBM Cloudant Cloud Architecture

Developers can build iOS/Android apps that allow users to:

- Connect to the Internet and replicate data for offline use
- Update data while offline
- Sync data when back online

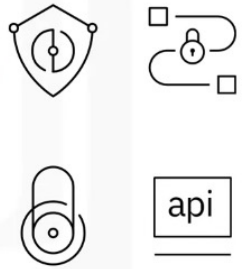


- Offline sync technology enables developers to build iOS and Android applications that allow users to: connect to the Internet and replicate data for offline use, update data when offline, and then sync the data when they are back online again.

IBM Cloudant Cloud Architecture

Security

- ISO27001, SOC 2 Type 2 compliant and HIPAA ready
- All data is encrypted, whether at rest or being transported on a network
- Optional user-defined encryption key management through IBM Key Protect
- Integrates with IBM Identity and Access Management (IAM) for granular access control at the API level



- In terms of security, IBM Cloudant is **ISO**27001, SOC 2 Type 2 compliant and **HIPAA ready.**
- All data is **encrypted,** whether that data is at rest or being transported across a network.
- There is also optional **user-defined encryption key management** on offer through IBM Key Protect.
- The service integrates with IBM Identity and Access Management **(IAM)** for granular access control at the API level.

IBM Cloudant Cloud Architecture

IBM Cloudant offerings

- A fully managed service
- An on-premises version
- And hybrid cloud deployment

IBM Cloudant Key Technologies

- Operational datastore – good fit for web/mobile apps

- Document-based NoSQL database – NOT a data warehouse

- Uses a well-defined HTTP API – fits into a modern, service-oriented architecture

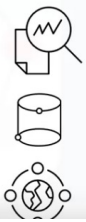
- HTTP API allows interactivity through browser or cURL – returning in JSON format

- IBM Cloudant provides several key technology components:
- As an operational data store, Cloudant is a good fit for any web or mobile application.
- **Cloudant is a document-based NoSQL database**, it is **not a data warehouse**, **like Hadoop** is.
- Cloudant uses a simple yet **well-defined HTTP API**, so it's a lot like working with a RESTful Web service.
- It's intended to fit into a modern architecture, and fits seamlessly into a service-oriented architecture without needing to build the abstracted layer.
- So you can deploy your database in the cloud of your choosing.
- The **HTTP API** lets **you interact** with the data through a browser or cURL **returning in the JSON** format.

IBM Cloudant Key Technologies

Fully integrated replication and sync processes

- MapReduce
- Full-text Search
- Geospatial



- Cloudant has **fully integrated replication and sync.**
- There are a number of different **ways to index and slice and dice** the data.
- **MapReduce is** a highly efficient way of crunching data in large data sets, and really good for doing **real-time analytics.**
- Search engines are very popular with web and mobile application developers.
- Many applications make **asynchronous calls** under the covers for doing **full text adhoc search** of your data.
- **Other DBaaS** offerings require you to **integrate with a third-party search engin**e, such as Elastic Search or Apache Solar which are both built on **Lucene** whereas, Cloudant has full text search fully built in to both the database-as-a-service and local offerings.
- **GeoSpatial** is ideal for **applications** involving **oil, gas, or transportation**, for example, satellites and transportation vehicles.

The IBM Cloudant Dashboard provides an easy way to monitor, manage, and develop your databases.

Summary

In this video, you learned that:

- Cloudant provides over 55 global data centers
- It is supported on several cloud platforms
- Its cloud architecture provides high availability, disaster recovery, optimal performance
- Cloudant does offline sync for web/mobile apps
- Cloudant can be offered as a fully managed service, an on-premise deployment, or a hybrid cloud deployment
- Cloudant consists of several key technology components - HTTP API, MapReduce, Full-text Search, Geospatial, Cloudant Dashboard

Cloudant Benefits and Solutions

- ❑ Describe the key benefits of IBM Cloudant
- ❑ Explain the challenges that IBM Cloudant can provide a solution to
- ❑ Describe common use cases for IBM Cloudant

The Benefits of IBM Cloudant

Scalability

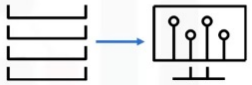
- IBM Cloudant scales to an enormous degree
  - Scales in size of data
  - Scales in number of concurrent users
- IBM Cloudant can be appropriate for:
  - A small startup with an app handling 1GB of data and 10 users
  - An enterprise organization with multiple apps handling petabytes of data and 20 million active users

- Cloudant scales to an enormous degree.
- It scales both in terms of data size and number of concurrent users supported.
- Cloudant can be appropriate for either; a startup company who is building an application handling 1 Gigabyte of data and 10 concurrent users, but it can also be an appropriate solution for an enterprise organization with multiple apps storing petabytes of data and 20 million concurrent active users.

The Benefits of IBM Cloudant

Availability

- An operational datastore for online apps



- Aims to be available 24 hours a day, 7 days a week, 365 days a year

- Cloudant allows for scaling up or down as required.
- Cloudant is always available. It's an operational datastore for online applications, so building a service that aims to be available 24x7x365 is a must.

Other Benefits of IBM Cloudant

- Cloudant is **durable,** aiming to never lose data, which it achieves by storing multiple copies of all data across separate physical nodes.
- Cloudant is **partition tolerant,** in order to meet the most stringent **high availability disaster recovery requirements**, whether handling node failures in a cluster, or even full data center outages.
- Cloudant provides **online upgrades** It is uniquely engineered to patch/upgrade on the fly…no outage necessary!
- **Clusters can be upgraded** without taking a **customer's database offline.**
- It provides online and offline access, which is ideal when you have scenarios with 'spotty' connectivity, for example, in remote areas and airplanes.

IBM Cloudant Provided Solutions

- Cloudant Customers rely on Cloudant to **solve many data layer challenges**.
- If your relational database isn't scaling the way you'd like - **Cloudant scales easily** for an exponentially growing user base.
- Developing and hosting your own databases has proven to be a challenge with limited budgets, time, and skill set for administration – Cloudant can help **reduce those costs**.
- Sometimes, you might be building an application from scratch without fully knowing the capacity requirements – Cloudant provides a simple and fast way to develop your applications.

IBM Cloudant Solutions

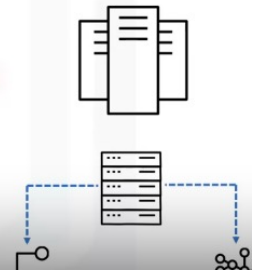
- Document database – logical collection of documents with single permission set
- IBM Cloudant documents are stored in JSON format – flexible schema
- Documents are stored in a database for two primary reasons:
  - Security access - apply access roles (read, write, admin, replicate) at the database level
  - Querying - you can't index or query within a single API call across databases. A cluster can hold any number of databases

- A **document database** is a logical collection of documents with a single set of access permissions.
- With Cloudant, documents are stored in the popular **JSON format** with **a flexible schema**.
- Documents **are organized** into databases for two primary reasons.
- This first is for **security access reasons**, as you can apply access roles (read, write, admin, replicate) at the database level.
- The second is **for querying**, as you can't index or query within a single API call across databases.
- Your cluster can hold any number of databases.

IBM Cloudant Solutions

When you sign up for a Cloudant account

- Physical servers do the work for you
- Run on cluster of servers containing load balancers and database nodes



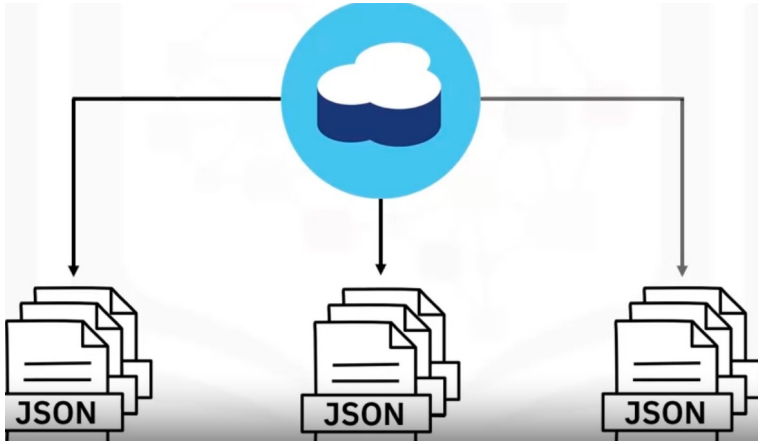
- When you sign up for a Cloudant account, there are actual physical servers set up that do the work for you.
- Cloudant runs on a cluster of servers consisting of load balancers and database nodes on the appropriate hardware

IBM Cloudant Solutions

- Cloudant is horizontally scalable – distribute database across a cluster
- Cloudant auto-shards (auto-partitions) data across a cluster
- Start with small number of nodes and let Cloudant run management scripts as your data grows
- Data is replicated to servers within the cluster to keep them in sync with each other
- Failed nodes are re-synced when back online

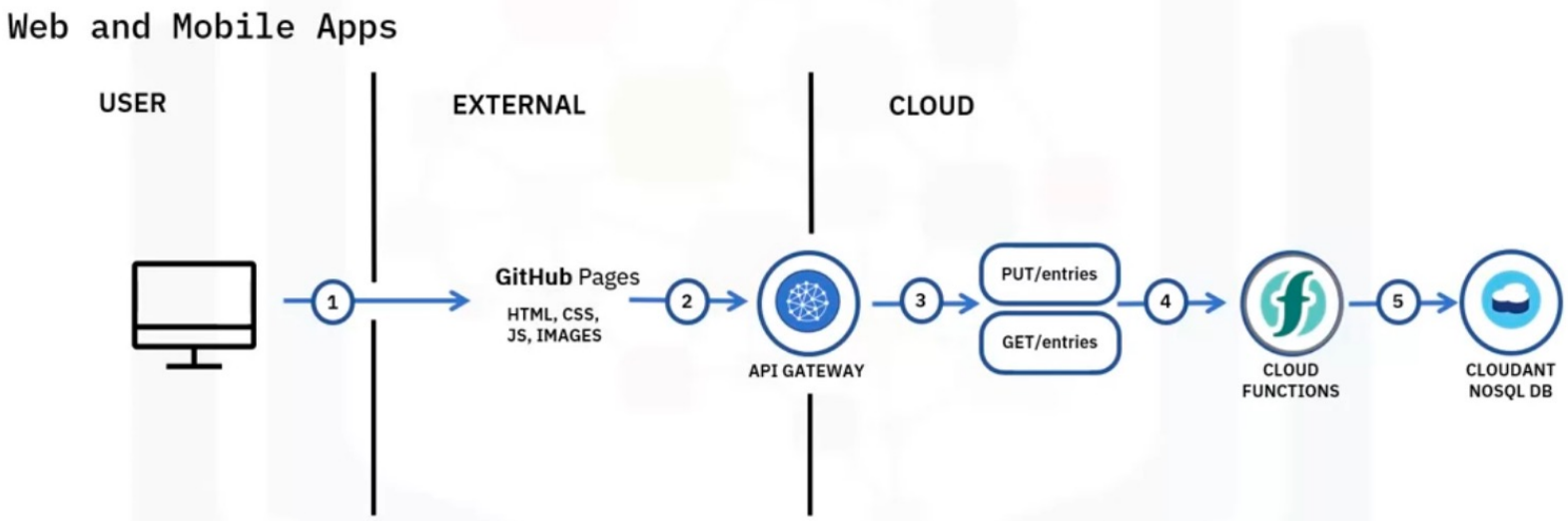
- Cloudant is horizontally scalable meaning you can distribute your database across a cluster.
- Cloudant auto-shards data across a cluster meaning that it auto-partitions the data evenly across clusters.
- So, you don't need to manually distribute the data as you would with a relational data store.
- You can start with a small number of nodes, and let Cloudant run scripts to manage your data as it grows to keep your data available.
- This means you don't have to do much capacity planning up front because Cloudant can easily adapt as your data needs increase and maintain performance.
- The data is replicated within the cluster to keep them synchronized without any administrative intervention.
- If a node fails and is then brought back online, it will automatically be brought up-to-date using replication to sync the data.

- To replicate across geos or clusters, you can **configure when and how** that replication will happen.
- If you're storing data across a cluster, it doesn't really do you any good to store one copy on one node because if the node fails, then you'll lose the data, and it will become unavailable to your applications.



- Cloudant employs a **quorum-style clustering** which stores every JSON document in triplicate across three separate physical nodes.
- When your application reads and writes data, Cloudant uses a **load balancer** that distributes the reads and writes evenly across the cluster, so the workload is distributed.
- So if one node fails, the data is still available on another node.

IBM Cloudant Case Studies

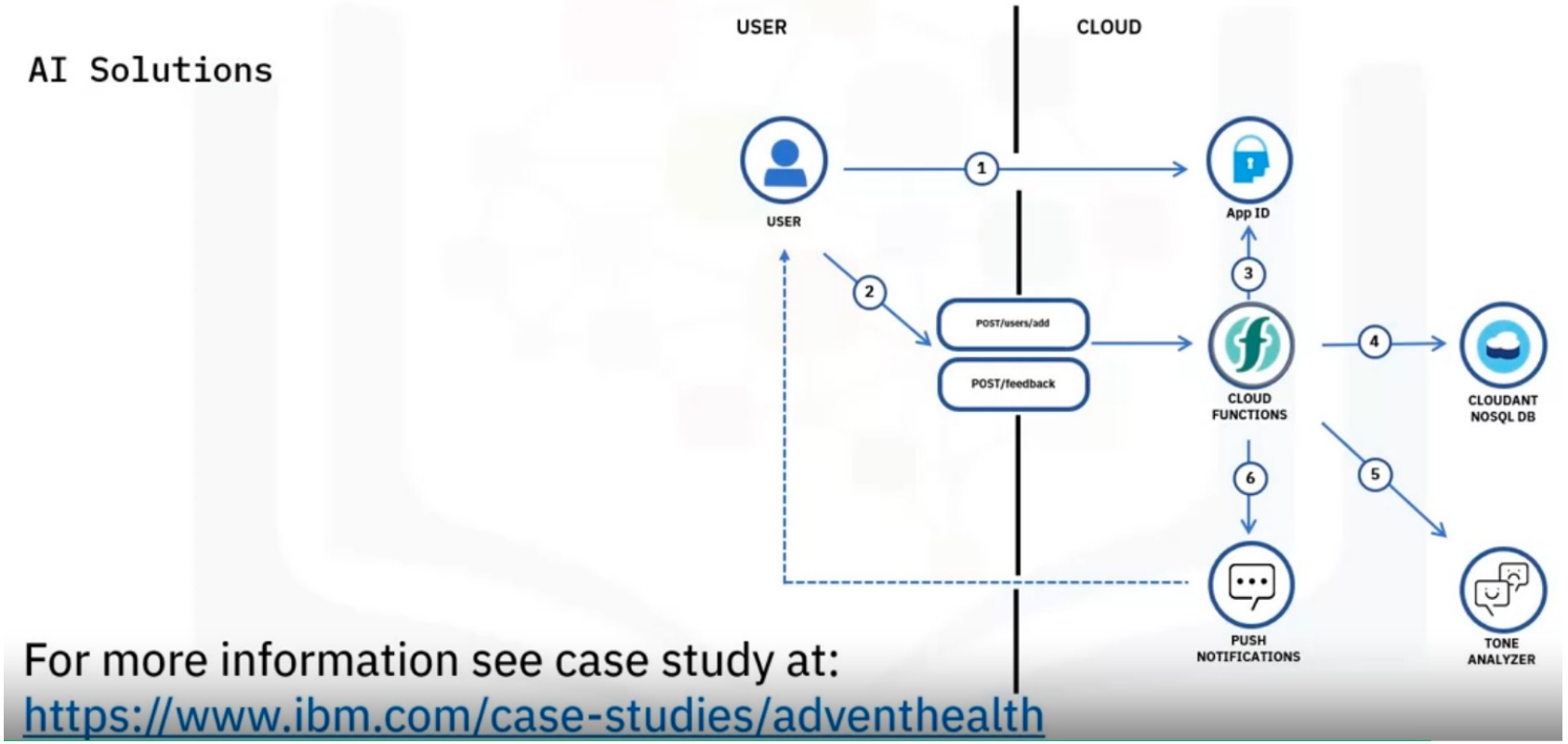


For more information see case study at:
https://www.ibm.com/case-studies/i944017z10766i92

- Cloudant can be used to build **web and mobile apps.**
- For example, you can build **point-of-sale mobile applications** using analytics to **personalize shopping experiences**.
- For more information, see the Quetzal case study at the URL listed on this page.

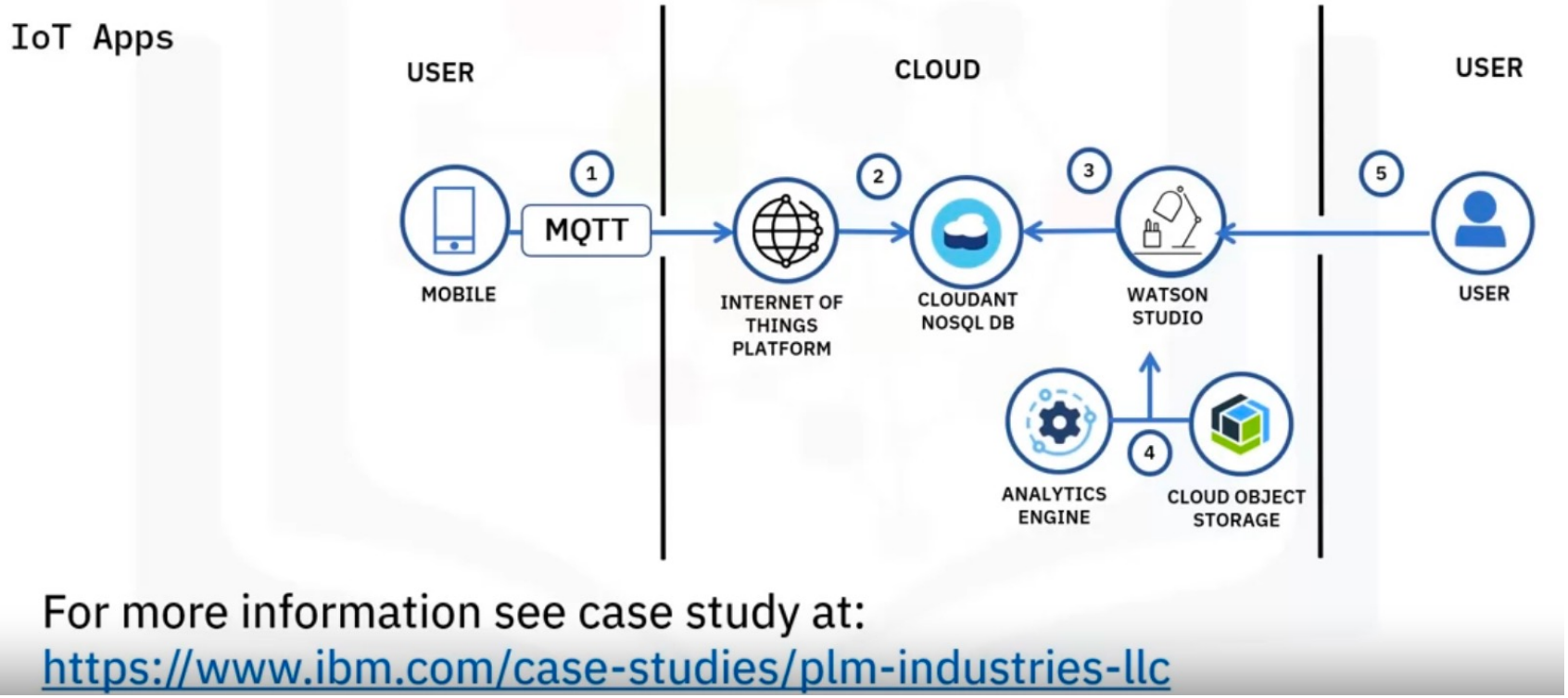
IBM Cloudant Case Studies

AI Solutions



- With Cloudant you can create powerful cognitive AI applications by connecting IBM Watson machine learning to data stored in Cloudant.
- For more information, see the Advent Health Partners case study at the URL listed on this page

IBM Cloudant Case Studies



For more information see case study at:
https://www.ibm.com/case-studies/plm-industries-llc

- With Cloudant you can analyze Internet of Things (IoT) sensor data stored in flexible Cloudant schemas, to trace end-to-end shipments and detect anomalies.
- For more information, see the PLM Industries case study at the URL listed on this page

Summary

In this video, you learned that:

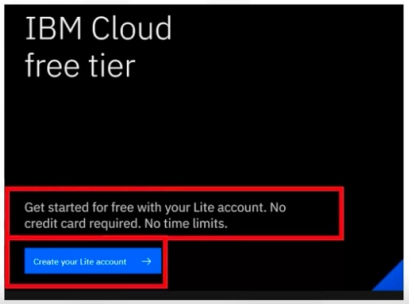
- Cloudant's key benefits are scalability, availability, durability, partition tolerance, and online upgrades
- Cloudant solves several data challenges including exponential user growth, increasing cost, time and skills required to develop, host and administer a database in-house
- Cloudant uses a document database for improved security and querying
- Cloudant runs on clustered servers in the Cloud
- Typical use cases for Cloudant include building web and mobile apps, AI solutions, and analysis of IoT data

Deployment Options for Cloudant

- ❑ Describe the different deployment plans for IBM Cloudant
- ❑ Describe the free Plan for Cloudant
- ❑ Describe the Standard Plan for Cloudant
- ❑ Describe the Dedicated Hardware Plan for Cloudant

Cloudant – Lite Plan

- Sign up for a Cloudant multi-tenant account at Cloudant.com
  - Use it for evaluation, prototyping, and development
  - No credit card required for sign up

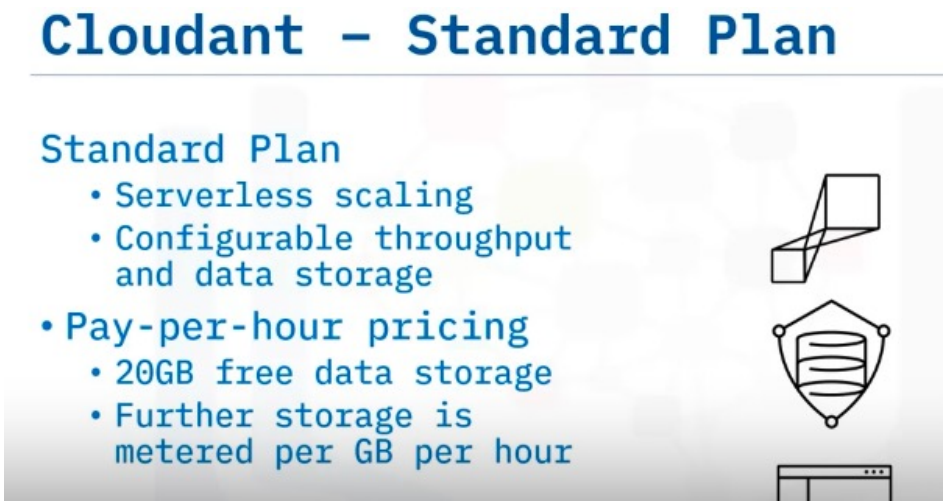


Cloudant - Lite Plan

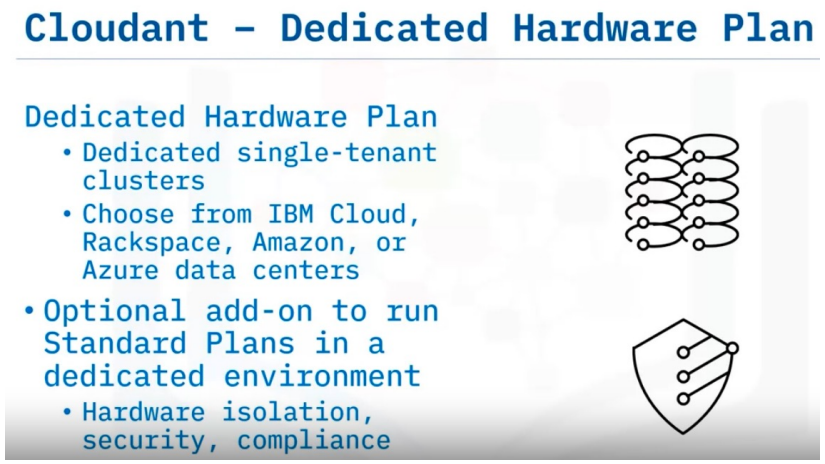
Lite Plan

- Perpetually free for evaluation and development
- Fixed limited amount of throughput capacity and data storage
- Provides full IBM Cloudant functionality
- Plan services deleted after 30 days of inactivity

- You can sign up for a free Cloudant multi- tenant account at Cloudant.com and then use it for evaluation, prototyping,and development.
- No credit card is required to sign up for this free plan, and there are no time limitations either.
- This free deployment version is called the Lite Plan.
- You can use the Lite Plan to take advantage of a perpetually free Cloudant offering for evaluation and development.
- The free Lite Plan has a **fixed limited amount** of provisioned **throughput capacity and storage**.
- For example, the provisioned throughput capacity is fixed at 20 reads per second, 10 writes per second and 5 global queries per second, there is also has a maximum size limit of 1MB for a single JSON document in this plan.
- However, the Lite Plan does provide access to the full functionality of IBM Cloudant for evaluation and development purposes. If you are inactive in your Lite Plan environment for a period of over 30 days, the services are deleted.

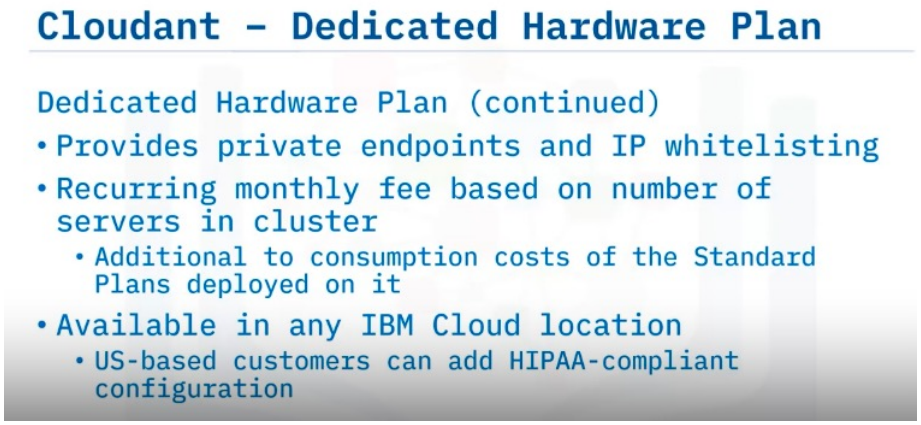


- If your organization wants a **fully- managed, but configurable**, database-as-a-service production deployment, you can opt for the Cloudant Standard Plan With the Cloudant Standard Plan, you get serverless scaling of throughput and data storage, which you can configure as your app needs change.
- You can add blocks of 100 Reads per second, 50 writes per second and 5 global queries per second.
- The Standard Plan offers pay-per-hour pricing, so you can control your costs.
- The hourly rate is charged on the throughput capacity allocated, not the metered volume of read and write requests.
- Included in the Standard Plan is 20GB free data storage; with any additional storage usage being metered at a defined cost per GB per hour.



- If your organizational needs **warrant a bare metal dedicated environment** for your fully-managed database-as-a-service production deployment, then you should opt for the Cloudant Dedicated Hardware Plan
- With the Cloudant Dedicated Hardware Plan, IBM **set up your Cloudant account** on dedicated, single-tenant clusters in the **IBM Cloud, Rackspace, Amazon, or Azure** data center of your choice.
- The Cloudant Dedicated Hardware plan is an optional add-on to run one or more of your Standard Plan instances on, and it provides a dedicated environment that benefits from hardware isolation, greater security, and better compliance.

- The Cloudant Dedicated Hardware Plan includes **private endpoints in addition to public endpoints,** and also offers IP whitelisting capabilities.
- The Cloudant Dedicated Hardware Plan is charged as a recurring monthly fee based on how many servers are in your cluster.
- The fixed price of this plan is in addition to the consumption pricing of the Standard Plan instances that you have deployed on it. It uses **daily prorated billing**, and has a **1-month minimum duration charge.**
- The plan is available in any IBM Cloud location, and any customers based in the United States can also take advantage of an optional HIPAA-compliant configuration for improved compliance.



Summary

In this video, you learned that:

- IBM Cloudant can be deployed using the Free Plan, the Standard Plan, or the Dedicated Hardware Plan
- Free Plan used for evaluation and development environments - limited throughput capacity and data storage
- Standard Plan provides serverless scaling of throughput and data storage - pay-per-hour basis
- Dedicated Hardware Plan setup on dedicated, single-tenant clusters - optional add-on provides isolated and secure dedicated environment

Summary and Highlights

Congratulations! You have completed this lesson. At this point in the course, you know:

•Cloudant is a fully managed DBaaS built on Apache CouchDB that uses a JSON document store.

•Cloudant is a distributed database optimized for large workloads, web, mobile, IoT, and serverless apps.

•Cloudant offers a powerful replication protocol.

•Cloudant's cloud architecture provides high availability, disaster recovery, and optimal performance.

•Cloudant can be offered as a fully managed service, an on-premises deployment, or a hybrid cloud deployment.

•Key technology components of Cloudant are HTTP API, MapReduce, Full-text Search, and Geospatial.

•Cloudant’s key benefits are scalability, availability, durability, partition tolerance, and online upgrades.

•Cloudant uses a document database for improved security and querying.

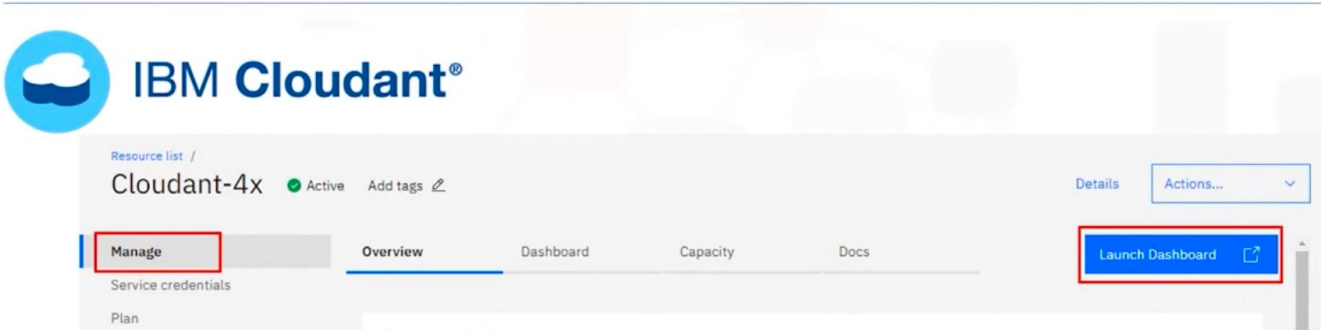
•Typical use cases for Cloudant include building web and mobile apps, AI solutions, and analysis of IoT data.

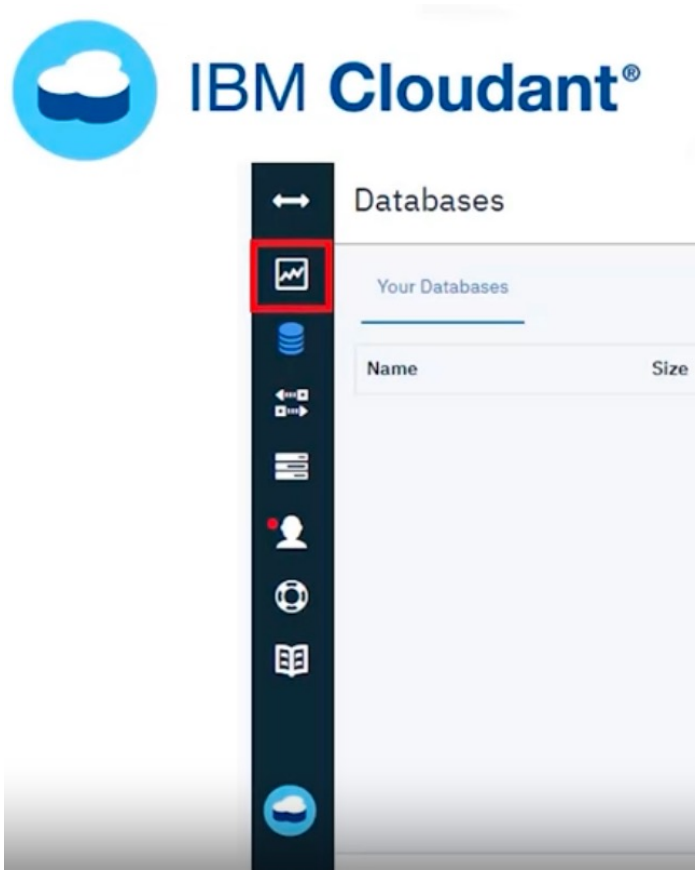
•IBM Cloudant has three deployment options

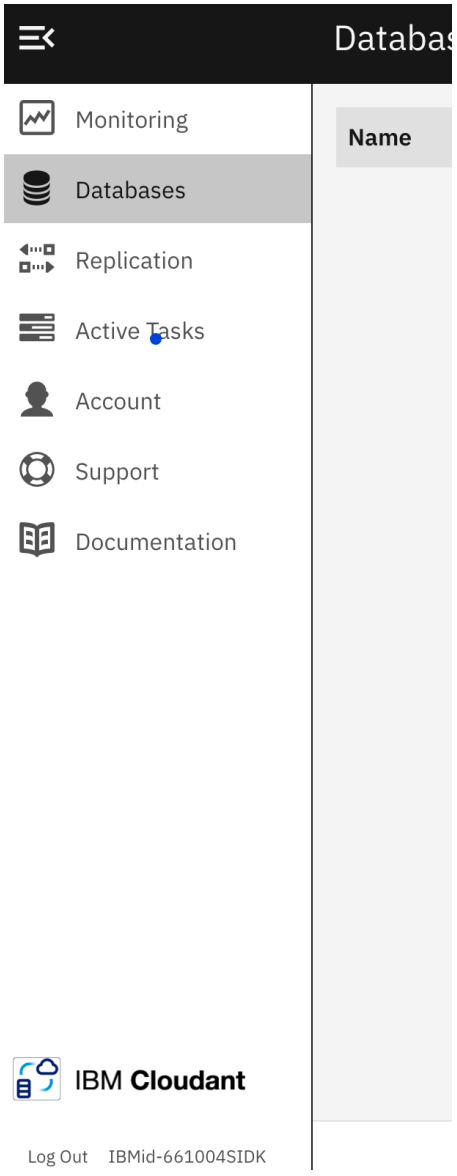
Part : 2 : Working With Cloudant

Dashboards in Cloudant

- ❑ Access the Cloudant dashboard.
- ❑ Describe the different elements that can be managed in the Cloudant dashboard.
- ❑ Use the Cloudant dashboard to manage a Cloudant service instance

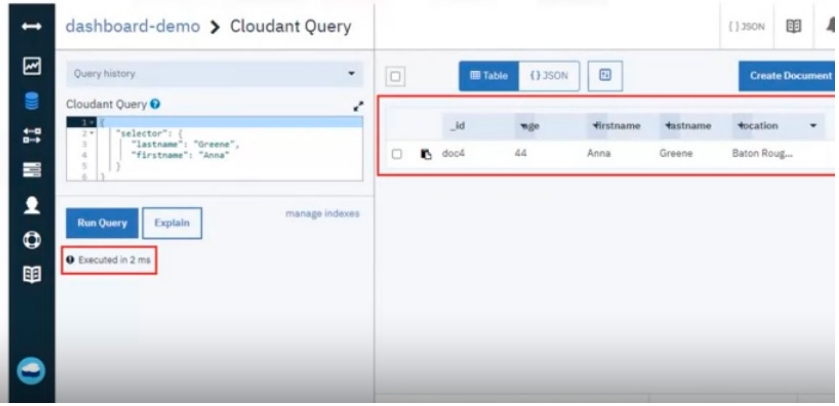



IBM Cloudant®
Databases
Your Databases
Name
Size


Databas
Monitoring
Databases
Replication
Active Tasks
Account
Support
Documentation
Name
IBM Cloudant
Log Out
IBMid-661004SIDK

Managing Databases in the Cloudant Dashboard

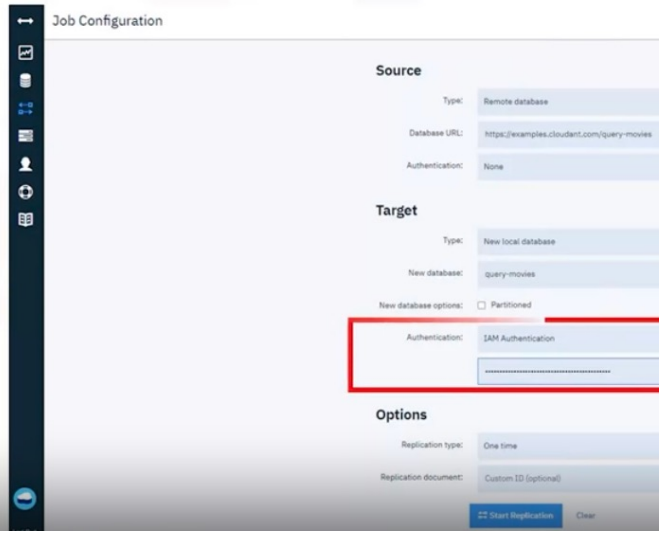
- Create databases
- Add documents to a database
- Replicate a database
- Set permissions on a database
- Delete a database
- Run queries on database documents



- In the database tab of the dashboard, you can: create databases, add documents to a database, replicate a database, configure permissions for a database, delete a database, and run queries on documents in a database.

Managing Replication in the Cloudant Dashboard

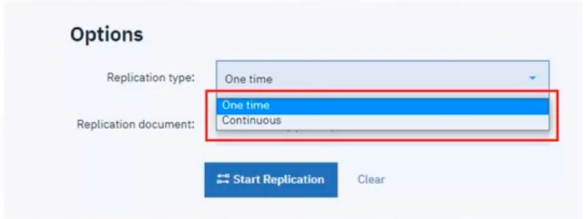
- Use Replication to configure replication between source and target databases
- Replication operations copy all changes from source to target databases
- Replication configuration:
  - Source database, target database, authentication, replication type



- You use the Replication tab of the Cloudant dashboard to configure replication between source and target databases.
- When you perform a replication operation it copies all changes from the source database to the target database.
- For example, if you add a document to, or delete a document from, the source database, after replication completes, that document will also be added to (or deleted from) the target database.
- When you configure replication, you need to specify several elements such as the source and target databases, the authentication used for the connection, and the replication type

Replication Types in the Cloudant Dashboard

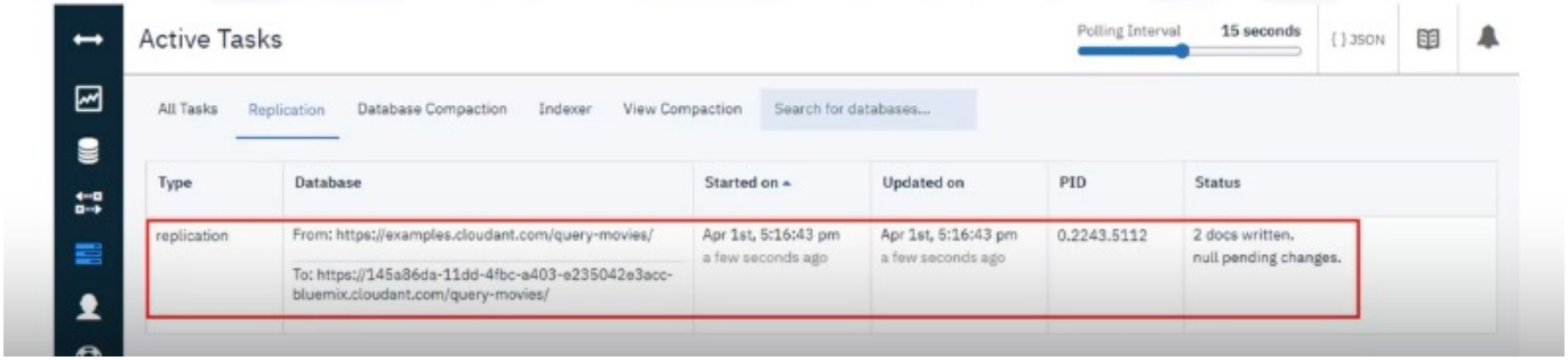
- One time
  - Occurs once after replication is configured
- Continuous
  - Changes replicated persistently until cancelled
- Replications have huge impact on performance
  - Continuous replication means impact will be constant
  - Continuous replication increases cost



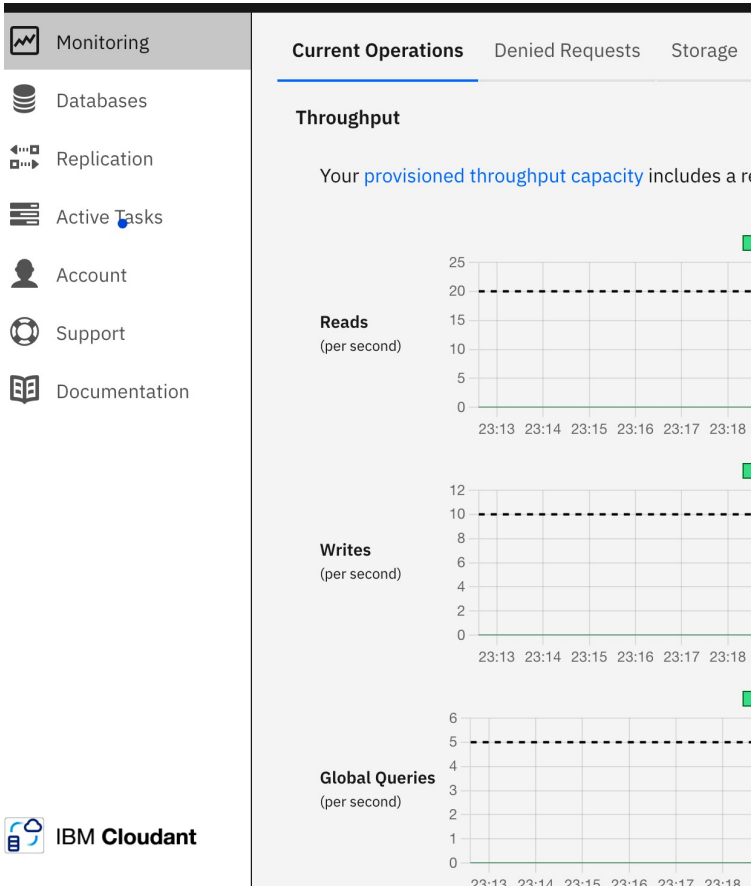
- Replication types are either ‘One time’, which means replication occurs once, immediately after replication is configured, but it does not continue to replicate after that time; which is the default replication setting.
- Alternatively, you can configure replication to use the ‘Continuous’ type of replication, where changes in the source database are replicated to the target database persistently until you specifically decide to cancel replication.
- Replications can have a huge impact on the performance of your Cloudant instance, so it is recommended to carry out some form of performance testing to identify any potential impact on your Cloudant environment.
- Furthermore, if you configure the ‘Continuous’ type of replication in your environment, then obviously that potential impact will be constant.
- Additionally, the extra system calls required to perform continuous replication will likely increase the cost to an organization running multi-tenant instances of IBM Cloudant.
- This is why continuous replication is not enabled by default.

Viewing Active Tasks in the Cloudant Dashboard

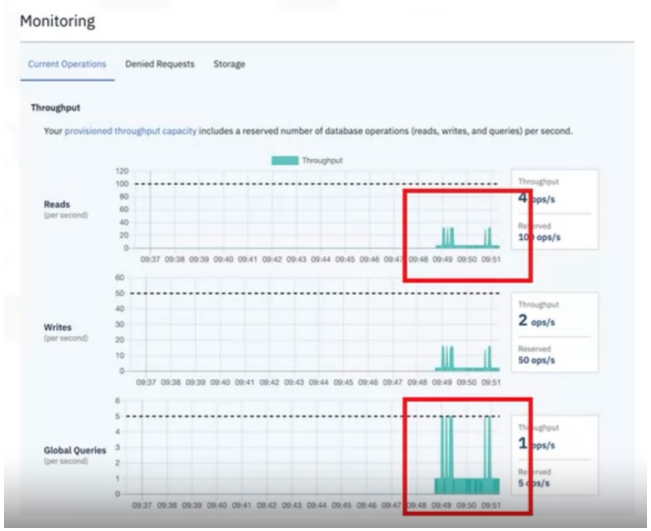
- Active Tasks – Replication, Compaction, Indexing



- Once replication has started, you can use Refresh to update the state to identify when replication has completed.
- When you go back to your Databases tab, you will see any replicated databases in the list.
- You can use the Active Tasks tab of the Cloudant dashboard to show a list of any tasks that are currently running, which may assist you in determining any issues you may have with system performance.
- The list of active tasks includes replication, compaction, and indexing processes.



- You use the Monitoring tab of the Cloudant dashboard to view the consumption of provisioned throughput capacity and data storage currently being utilized by your applications in your Cloudant instance.
- On the three tabs available in the Monitoring section, you can monitor: Current Operations – these graphs show minute-by-minute consumption of your provisioned throughput capacity.
- This throughput capacity is broken up into reads per second, writes per second, and global queries per second.
- The dotted lines indicate the peak capacity limit allowed by your Cloudant instance.
- In your hands-on labs, you will be using the Lite Plan and therefore you will not see any data in these graphs on this page, as displayed in the screenshot on the left.

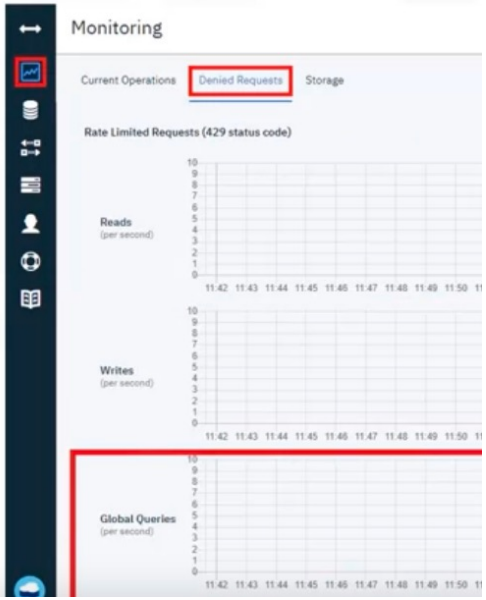


This is how it might look in production environment.

Monitoring with the Cloudant Dashboard

Denied Requests

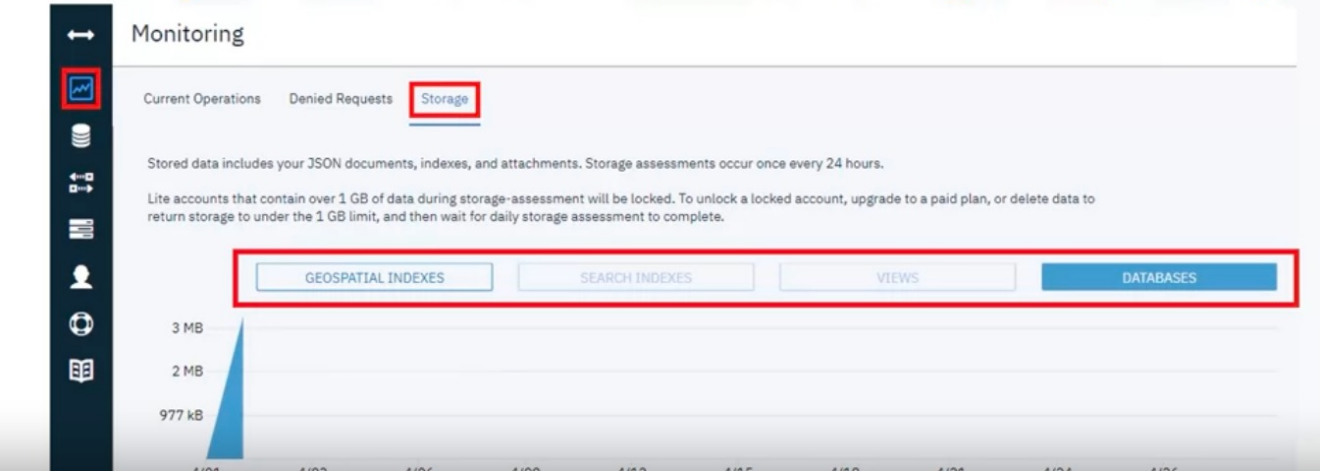
- Minute-by-minute monitoring
- *429: too many requests*
- When throughput capacity is exceeded



Monitoring with the Cloudant Dashboard

Storage

- Daily record of current storage capacity



- The Storage graphs display a color-coded daily record of the current storage capacity used by four elements of your Cloudant instance – namely Geospatial Indexes, Search Indexes, Views, and Databases.
- The first screenshot is currently only displaying information for the Databases element.
- You can select each colored bar above the graph to toggle the display of each element on or off.

Monitoring with the Cloudant Dashboard

Storage

- Daily record of current storage capacity

- The second screenshot is configured to display information for all four elements.
- You can use the information displayed in these graphs to help you determine when capacity is being reached for an element of your Cloudant instance, in case you need to modify your capacity settings.

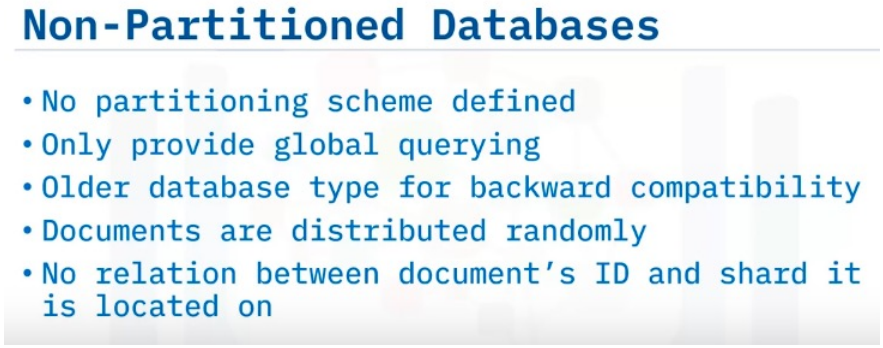
Summary

In this video, you learned that:

- You use the Cloudant dashboard to create and manage databases, configure replication, view active tasks, and monitor Cloudant
- You use the Databases tab to create databases, add documents to databases, replicate databases, configure permissions, delete databases, and run queries on documents
- You use the Replication tab to configure replication between source and target databases
- You use the View Active Tasks tab to view list of currently running tasks
- You use the Monitoring tab to view the consumption of provisioned throughput capacity and data storage

Working with Databases in Cloudant

- ❑ Explain the difference between partitioned and non-partitioned databases
- ❑ describe how to create databases in Cloudant
- ❑ describe the characteristics of documents in a Cloudant database
- ❑ describe how to insert, view, query, update, and delete documents in a Cloudant database



- You can create and use **two types** of database in IBM Cloudant – non-partitioned and partitioned.
- **Non-partitioned** databases have no partitioning scheme that needs to be defined, which makes them relatively **simple to use**.
- However, non-partitioned databases **only provide global querying**.
- A non-partitioned database is a **legacy type of Cloudant database**, that is used with **earlier versions** of IBM Cloudant and also with CouchDB.
- In non-partitioned databases, **documents are distributed to shards ra**ndomly based on a transformation of their document ID.
- For this reason, there is no actual relation between a document's ID and the shard that it ends up being located on.
- In fact, it is unlikely that documents that share similar document IDs will be located on the same shard as each other.

Partitioned Databases

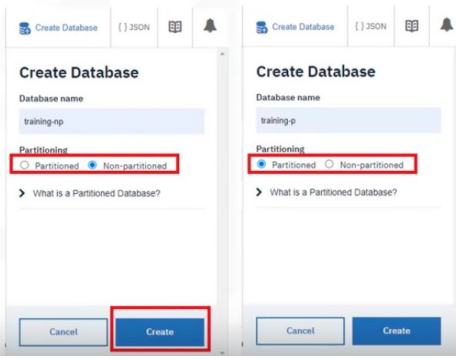
- Provide partitioned and global querying
- Offer performance and cost benefits
- Newer database type
- Require specification of logical partitioning
- Documents are assigned to a partition using a partition key
- Partitioned databases are highly recommended

IMPORTANT: You cannot change the partition type after database creation

- Partitioned databases provide **both partitioned and global querying.**
- They also offer substantial **performance and cost benefits.**
- Partitioned querying takes advantage of the data layout within the database cluster to deliver **improved** and more scalable **query performance.**
- **Partition queries** are also often **cheaper** than global queries.
- Partitioned databases are a newer type of Cloudant database. However, they **require** you to **specify a logical partitioning** of your data.
- A partitioned database **requires** a partition key for every document.
- A partition is a logical grouping of documents. All documents are assigned to a partition, and **several documents** are typically **given the same partition k**ey.
- It is highly recommended that you **use partitioned databases to get the best long-term performance** out of your databases, if your data model allows for logical partitioning of documents.
- **IMPORTANT: You cannot change the partitioning type of an existing database.**

Creating Databases in Cloudant

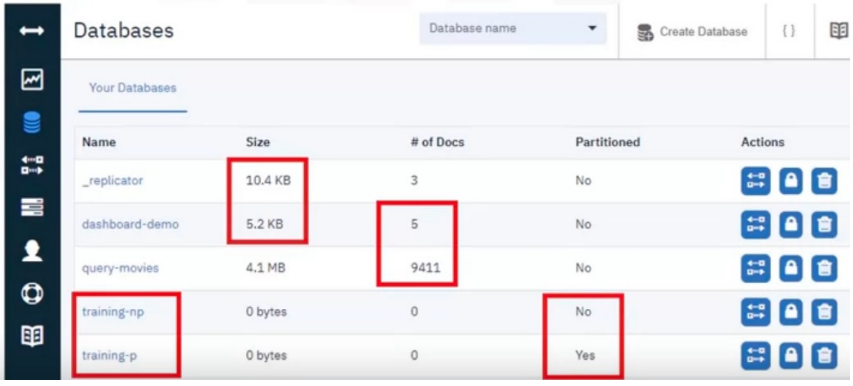
Using the dashboard:
- Create new database
- Select partitioning type
- Create



- You create new databases in Cloudant from the Databases dashboard tab.
- When you create a database, you must select whether you want the database to be partitioned or non-partitioned

Creating Databases in Cloudant

- Databases are listed on the Databases dashboard tab
- Displays:
  - Name
  - Size
  - # of documents
  - Partitioned (Yes/No)

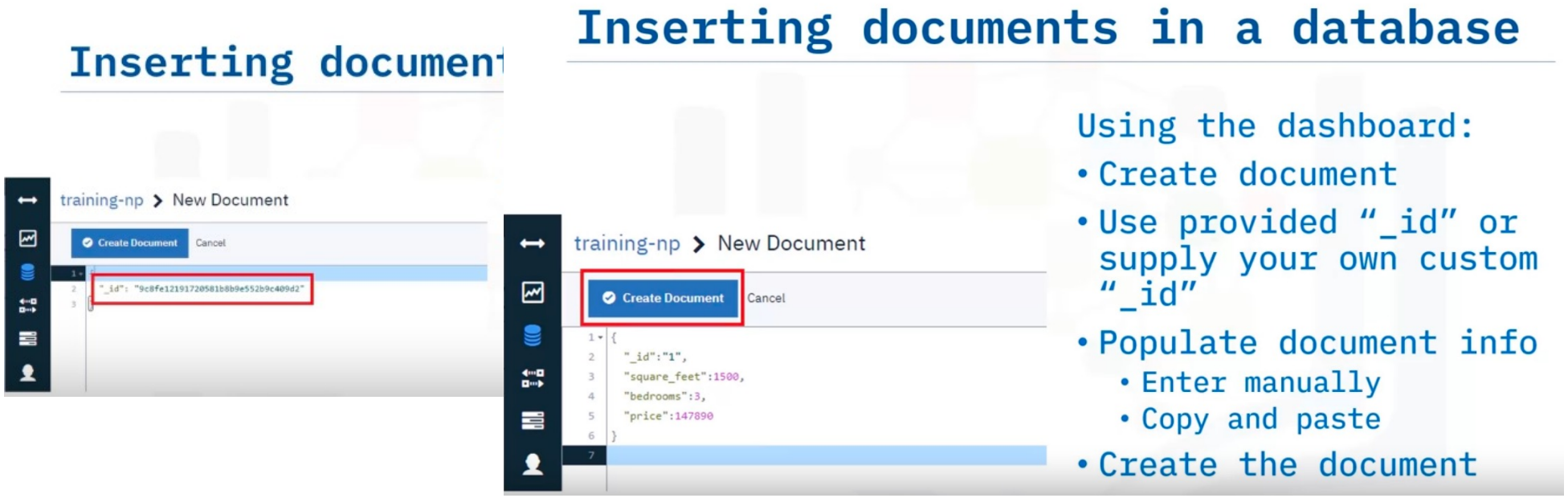


- Databases that have been created are displayed on the Databases dashboard tab.
- The list displays: Name information, the size of each database, the number of documents contained in each database, and whether the database is partitioned or not.

Documents in a database

- Other databases may store data in tables
  - Data contained in rows with the same fixed columns
  - Schema is predefined
- Cloudant is a document database system
  - Uses a JSON document store
  - Collections = databases in Cloudant (not tables)
  - Databases contain a number of documents
- Cloudant documents are JSON objects
  - Start with { and end with }
  - Contain a number of key:value attributes

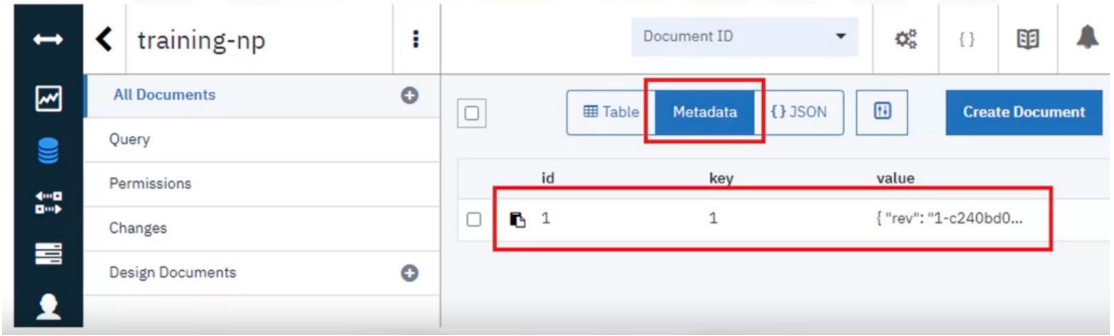
- Many other database systems store their data in tables, with data contained in rows, each with identical, fixed columns.
- And the **schema of** each **table is predefined**: a list of columns with their name, date type, value constraints, and relations to other tables carefully defined.
- But in Cloudant things are very different. We've already discussed that Cloudant is a document database type, that uses a JSON document store.
- So, your Cloudant instance has **collections called databases** (**rather than tables**), and each of these will contain a number of documents.
- A Cloudant document must be a **JSON object**, which must start and end with curly braces (or brackets) and it will contain a number of **key-value attributes**.
- The **JSON objects** must be **less than 1mb** and can contain any number of **strings, numbers, booleans, arrays, and nested objects**.



- You insert documents into a database on the details page for the selected database in the Cloudant dashboard.
- To insert a document, you need to create a document in the database.
- When inserting a document into a database you need to use the starting and ending curly braces, and you need to either use the provided "_id" or supply your own custom "_id".
- Cloudant uses the "_id" key to uniquely identify a document – it's the equivalent of the Primary Key in relational databases.
- You then populate the document details either by entering them manually between the curly braces or by pasting in from another source.
- Then you create the document to insert it into the database.

Viewing documents in a database

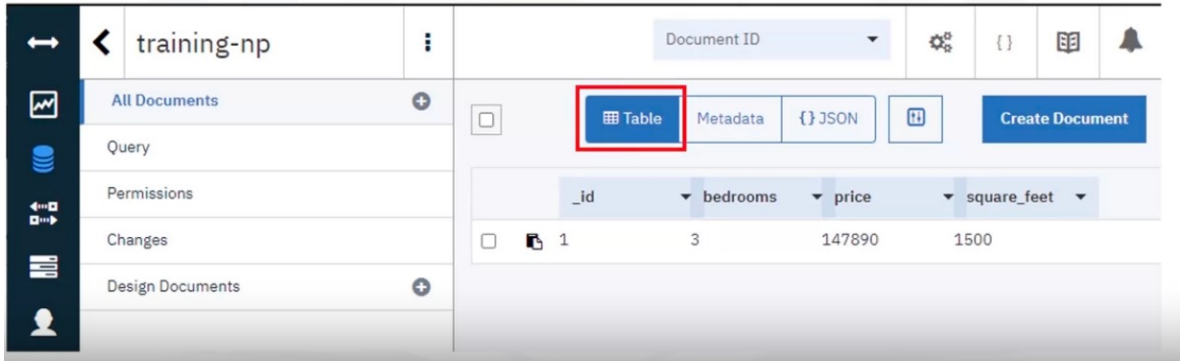
- Metadata view



- When you select a database, you can view the documents it contains by three different views.
- The first is Metadata view, which is the default view after you insert documents into a database, and this view shows ID, key, and value information for each document.

Viewing documents in a database

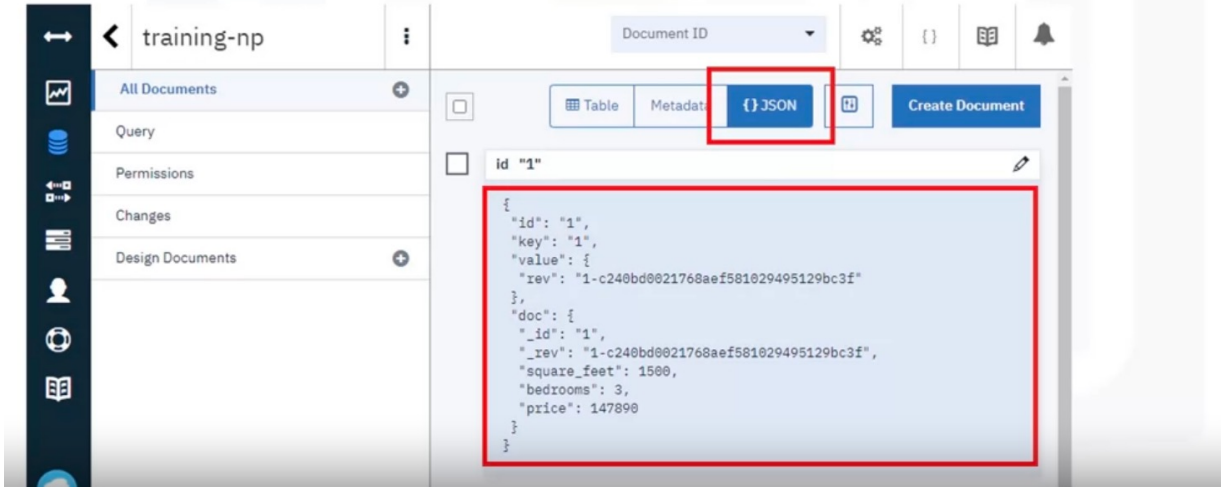
- Table view



- You can also view documents in a database in Table view, which shows information about the data tables for each document in a database.
- In this example we can see table data for ID, number of bedrooms, price, and square footage.

Viewing documents in a database

- JSON view



- And the last view you can use is JSON view, which shows the contents of the document in JSON format.
- You can also edit the JSON document from here.

- You can run queries against the documents in your database.
- You enter your query in the query window, then run the query.
- You can update the data in a document in your database.
- You simply select a document, make the changes you want, and then save your changes.
- You can delete documents from your databases using the Cloudant dashboard.
- You simply select the document, delete it, and confirm the deletion operation

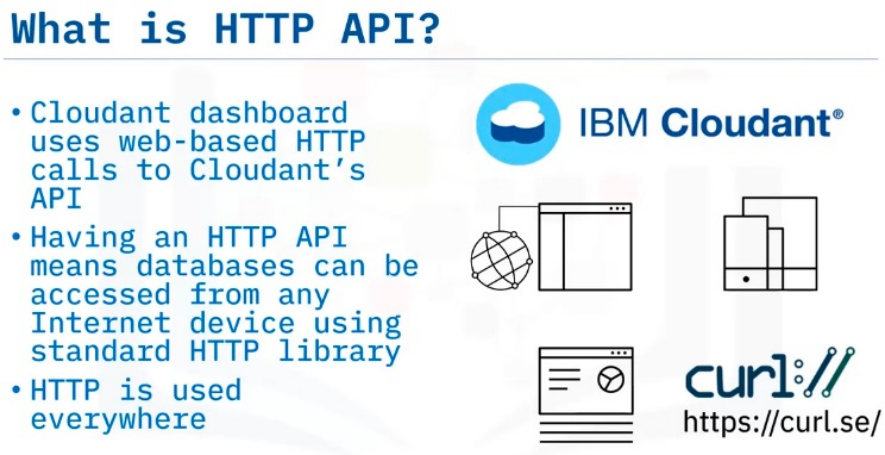
Summary

In this video, you learned that:

- There are two types of database in Cloudant – non-partitioned and partitioned
- Non-partitioned databases have no partitioning scheme to define, but they only provide global querying
- Partitioned databases provide partitioned and global querying, and offer performance and cost benefits
- Partitioning type is set at database creation
- Cloudant uses a JSON document store
- You can use the Cloudant dashboard to create databases
- You can use the Cloudant dashboard to insert, view, query, update, and delete documents in a database

HTTP API Basics

- ❑ Describe what the HTTP API is
- ❑ explain what curl is
- ❑ use simple curl commands.



- In an earlier video and lab, you saw how to use the Cloudant dashboard to perform basic CRUD operations in a Cloudant database.
- The Cloudant dashboard itself is a web-based app that makes **Hypertext Transfer Protocol (or HTTP)** calls to the **Application Programming Interface (or API)** in Cloudant.
- Because Cloudant's databases have an HTTP API, they have the substantial advantage that they can be accessed from any Internet device that wants **to read** data from them, **or write data** to them, and they can do this by **using requests** from the **standard HTTP library.**
- A device does not require specialized software, or drivers, or proprietary protocols, or anything of that nature, in order to work with Cloudant's databases.
- The following kinds of devices or technologies can **communicate** successfully using standard HTTP: Web browsers, mobile devices, common programming languages, and command-line **scripting tools, such as curl**
- In this video you will get an overview of the curl command-line and scripting tool to see how to make those HTTP calls (or requests), so that you can use it to perform the same CRUD operations you performed earlier in the course using the dashboard.

What is curl?

- Free, open source, command-line tool makes HTTP requests
- Part of Client URL (cURL) project
- Get and send data using URL syntax
- Also supports secure HTTP (HTTPS)
- Latest source code – *github.com/curl/curl*

curl://
https://curl.se/

- curl is a free, open source command-line tool that you can use to make HTTP requests to your Cloudant API.
- It is part of the 'Client URL' or 'cURL' computer software project that was originally released in 1997.
- Curl can **get and send data**, including files, using standard URL syntax directly from the command-line or for more repetitive tasks you could use it in scripts instead.
- It also **supports secure HTTP (or HTTPS**) requests by **using** Secure Sockets Layer (or **SSL**) certificate verification.
- You can access the latest curl source code at github.com/curl/curl

- If you just want to retrieve a web page, from the curl command-line prompt you simply enter the word **'curl' then the URL of the web page.**
- The example on this screen will retrieve the IBM Cloud home page.
- To stop you having to enter the URL of your Cloudant service every time you connect, you can create a variable.
- The example on this screen uses the 'export URL' command to save our Cloudant URL as a variable called URL.
- You need to **replace the 'username', 'password', and 'host'** parts **with your own information** from your Cloudant service credentials

Simple curl commands

To retrieve a web page:

```
curl https://www.ibm.com/cloud
```

To create a variable for Cloudant's URL:

```
export URL="https://username:password@host"
```

Simple curl commands…continued

To create an alias:

```
alias acurl="curl -sgH 'Content-type: application/json'"
```

To test connectivity:

```
acurl $URL/
```

To view a list of all databases:

```
acurl $URL/_all_dbs | python -m json.tool
```

- You could also create an alias (that is, a shortcut) for curl.
- The example on this screen will create an alias called '**acurl'** and **specifies the JSON content-type header.** It also uses some useful command-line switches.
- The lowercase **'s'** makes your **request silent** (that is, you won't see any progress or error messages returned), lowercase **'g' disables the URL globbing parser,** and uppercase **'H '**allows you to specify the **content-type header**.
- To test connectivity to your database in Cloudant, you can use the acurl alias and the URL variable.
- The variable must be preceded by a dollar sign.
- If you are successful you will get some JSON code returned, that includes a "Welcome" message, a "version" number, a "vendor" name, and "features" information about your Cloudant instance.
- To **retrieve a list** of all your **databases,** you again use the acurl alias and the $URL variable, but with the '_all_dbs' endpoint.
- As an additional option, if you have Python installed on your operating system, you can add the 'pipe' command at the end of this line.
- This sends the output of your command to **Python for improved JSON formatting** in the terminal, to transform the JSON data into a more readable format.

- There is also another JSON formatter called 'jq' which you can freely download and
- install, which will do a similar job of formatting JSON data into a more readable form.
- The pipe command for specifying this JSON formatter is '| jq'

To view a list of all databases:

```
acurl $URL/_all_dbs | jq
```

Simple curl commands…continued

To view the details for a single database:

```
acurl $URL/training | python -m json.tool
```

To view the documents in a database:

```
acurl $URL/training/_all_docs?include_docs=true
```

To retrieve a single document:

```
acurl $URL/training/<document id>
```

- If you want to view the details of a single database, just specify the name of the database after the forward slash.
- The example on this screen will retrieve the details of a database called ‘training’.
- Again, you can use one of the two optional ‘pipe’ commands to send the output of your command to Python or .jq for improved readability in the curl terminal.

- To view the documents contained in a database, just specify the name of the database after the forward slash and then the ‘_all_docs’ endpoint.
- This will retrieve only the ‘_id’ and ‘_rev’ values for all the documents.

- To retrieve the bodies of the documents too, you specify ‘?include_docs=true’ at the end of the command, as seen in this on-screen example.
- To retrieve a single document from your database, you specify the document’s document ID after the name of the database it’s contained in.

Summary

In this video, you learned that:

- Cloudant's databases have an HTTP API
- HTTP is used by web browsers, mobile devices, common programming languages, and command-line scripting tools
- Curl is a free, open source command-line tool to make HTTP requests to the Cloudant API
- Curl can get and send data using standard URL syntax
- You should create a variable to access Cloudant
- You can use an alias as a shortcut for curl
- You can pipe the output of your curl commands to Python or jq for improved JSON formatting

Working with the HTTP API

- ❑ Describe HTTP methods used in curl,
- ❑ use the HTTP API with databases,
- ❑ use the HTTP API with documents,
- ❑ use the HTTP API with Cloudant Query,
- ❑ use the HTTP API to run queries.

HTTP methods in curl

- Default HTTP method = GET (i.e.,retrieve)
- Other HTTP methods (verbs):
  - PUT = create a database, create a document or modify an existing document (use with document ID)
  - POST = run a query, create an index, create a document (use with -d switch)
  - DELETE = delete a database, or document, or index
- Curl uses '-X' to specify the HTTP method
  - Example: curl -X PUT

- In the previous video, you looked at some commands to run in curl to perform common tasks.
- You may have noticed that when we ran the last few curl commands, we didn't have to specify what HTTP method we were using with the commands.
- That is because when we run a command in curl without a specific method, it **defaults to using the 'GET' HTTP** method.

- The API in Cloudant typically uses **verbs** for its HTTP methods, so GET retrieves data for example.
- But what if we want to create a database, or add a document, or delete a document?
- These tasks require the use of other HTTP methods, such as **PUT** to **create a database, create a document, or modify an existing document** (with documents you use the document ID with your PUT command).
- **POST** to **run a query, create an index, or create a document** (with the POST method you need to use the **'–d'** command-line switch with your documents).
- **DELETE** to **delete databases, documents, and indexes**.
- However, **to use an HTTP method other than GET,** you need to use the uppercase '-X' command-line switch, and **then specify which method** you wish to use for the command in curl.
- For example: **curl –X PUT**

Using HTTP API with databases

Creating a database:

```
curl -X PUT $URL/training
```

Dropping a database:

```
curl -X DELETE $URL/training
```

Expected response:

```
{"ok":true}
```

- To create a database in Cloudant, you use the uppercase 'X' switch with the PUT HTTP request, then specify the service connection variable, and the name of the database you want to create.
- To delete a database (or the term more often used is to 'drop' a database) in Cloudant, you again use the uppercase 'X' switch, but this time with the DELETE HTTP request, then specify the service connection variable, and the name of the database you want to drop.
- To verify that either of these HTTP requests was successful, look for a response of "ok":true

Using HTTP API with documents

Inserting a document:

```
curl -X PUT $URL/training/"212" -d '{"coursename":"Excel Basics",
"level":"beginner"}'
```

Expected response:

```
{"ok":true, "id":"212","rev":"1-123456"}
```

- To insert a document in Cloudant, as before, you use the uppercase 'X' switch with the PUT HTTP request, then specify the service connection variable, and the name of the database you want to insert the document in.
- You **follow** this with **the document ID** for the document, which in this case you are manually specifying (in this example it is **"212"),** which is then followed by the "–d" command-line switch which allows you to enter the JSON data for the body of the document (in this example it is a course name and level).
- To verify that this PUT request was successful, look for a response of "ok":true, followed by the document id number you supplied, then a revision token number (or rev) which is a token value generated by the database when the document was written to it.

Using HTTP API with documents

Updating a document:

```
curl -X PUT $URL/training/212 -d '{"coursename":"Excel Basics",
"level":"beginner","edition":"third","_rev":"1-123456"}'
```

Error response:

```
{"error":"conflict","reason":"Document update conflict."}
```

Expected response:

```
{"ok":true, "id":"212","rev":"1-234567"}
```

- In order to update a document in one of your databases you **need** the document's **revision token value.**
- This can be obtained by either noting it down from the response when you create the document, or by using the "_all_docs" endpoint, as you saw in the previous video, to retrieve a list of all documents in a database.

- To update a document in Cloudant, as before, you use the uppercase 'X' switch with the PUT HTTP request, then specify the service connection variable, and then the path to the document you want to update, which is the database name followed by the document's ID.
- You follow this with the "–d" command-line switch, and then provide the new updated body of the document you are updating.
- In this example we have added an edition value to the document.
- It is important to note here that you must supply the entire body of the updated document – that is, **you cannot update part of a document** – you **must replace it in its entirety** with the updated version.
- Lastly, you **must provide** the **revision token** value for the document.
- If you do not provide this revision token value, you will see **an error message**, similar to the example shown here.
- To verify that this update request was successful, look for a response of "ok":true followed by the document id, followed by the new revision token value.

Using HTTP API with documents

Deleting a document:

```
curl -X DELETE $URL/training/212?rev=1-234567
```

Error response:

```
{"error":"not found","reason":"deleted"}
```

Expected response:

```
{"ok":true}
```

- To delete a document from a database, you use the uppercase 'X' switch, with the DELETE HTTP request, then the service connection variable, the path to the document you want to delete, and finally the current revision token value for the document.
- If you do not provide the correct revision token value, you will see an error message as shown here.
- To verify that this DELETE request was successful, look for a response of "ok":true.

- So far, all the operations you have seen have been CRUD operations (that is - create, read, update, and delete operations).
- But what if you want to find out information contained in the documents in your database?
- Such as which courses are for beginners, or which courses relate to a specific technology or software application, or which courses list Jane Doe as the instructor?
- This is where querying comes into the picture.
- **Cloudant Query's language is based on the MongoDB query lang**uage, which uses queries expressed in JSON formatted objects.
- These use a "selector" attribute to define the subset of data to be returned by the query.
- This is the equivalent of the WHERE clause in SQL.

HTTP API with Cloudant Query

Finding information in your documents:
"Which courses are for beginners?"
"Which courses cover Excel?"
"Which courses is Jane Doe an instructor for?"

- Cloudant Query is a query language based on MongoDB
- Queries expressed as JSON objects
  - Uses mandatory 'selector' attribute (SQL = WHERE)

Using HTTP API to run queries

- Running a query to find 'beginner' courses:

```
curl –X POST $URL/training/_find \
-H "Content-Type: application/json" \
-d '{"selector":{"level":"beginner"}}'
```

- Running a query to find courses by 'Jane Doe':

```
curl –X POST $URL/training/_find \
-H "Content-Type: application/json" \
-d '{"selector":{"instructor":"Jane Doe"}}'
```

- A JSON query object is sent using the 'POST' HTTP request to the database's "_find" endpoint to perform the query.
- These JSON queries can be triggered form the curl command-line, or from the Cloudant dashboard.
- The example on this screen is using curl to look for a document with a document ID of 212.
- Note the use of the 'selector' attribute.
- The uppercase **"-H"** switch allows you to specify the **JSON content-type header**.
- The examples on this screen are looking for all courses that have a level of "beginner", and all courses that have "Jane Doe" listed as the instructor.

Using HTTP API to run queries

- Running a query with logical operators:
  - $gt = greater than
  - $lt = less than

```
curl –X POST $URL/training/_find \
-H "Content-Type: application/json" \
-d '{"selector" {"modules":{"$gt":4}}}'
```

- You can also use logical operators in your queries, such as greater than - which is "$gt", and less than - which is "$lt".
- So the example on this slide is looking for courses that have more than 4 modules in them.

Using HTTP API with Cloudant Query

- For ease of reuse and for large JSON queries
  - Store your JSON query in a .json file
  - Specify the JSON file using '-d@*jsonfilename*'
- Running a query using a .json file:

```
curl –X POST $URL/training/_find \
-d@myquery.json
```

- If you have a query that you want to reuse regularly, or the JSON query is quite large, rather than entering your JSON query directly in the curl command-line as we have done here and in the hands-on lab, you will likely want to store your JSON query in a .json file instead.
- If you do this, you specify the JSON file using '- d@jsonfilename', as seen in the example on this screen.

Summary

In this video, you learned that:

- Curl defaults to 'GET' HTTP method unless another method specified by "-X" command-line switch
- Other HTTP methods you can use in curl are; PUT, POST, and DELETE
- You use the PUT request to create databases, create documents, and update documents
- You use the POST request to run queries, create documents, and create indexes
- You use the DELETE request to delete (or drop) databases, documents, and indexes
- Cloudant Query is a query language based on MongoDB's to run queries expressed as JSON objects

How to Access Documentation and Support Resources

There is a plethora of further information about IBM Cloudant online, including documentation, training content, and support resources.

Here are some useful direct links to those resources:

•Documentation – https://cloud.ibm.com/docs/Cloudant

•Training Content – https://cloud.ibm.com/docs/Cloudant?topic=Cloudant-learning-center

•Support Resources – https://cloud.ibm.com/docs/get-support

Summary and Highlights

Congratulations! You have completed this lesson. At this point in the course, you know:

•You use the Cloudant dashboard to create and manage databases, configure replication, view active tasks, and monitor Cloudant.

•You can access your Cloudant service instance on the IBM Cloud dashboard from the Resources list in the Services list.

•You use partitioned databases to get the best long-term performance out of your databases if your data model allows for logical partitioning of documents.

•When you create an IBM Cloudant database, you must select whether you want the database to be partitioned or non-partitioned.

•A Cloudant document must be a JSON object that contains key-value attributes.

•You can view your database documents by three different views (Metadata, Table, JSON).

•You can use the curl command line tool to make HTTP requests to the Cloudant API.

•You specify an HTTP method to create a database, add a document, or delete a document.

•To update a document in one of your databases, you need the document's revision token value.

•You can run a querying operation to find out information about the documents in your database or to retrieve a subset of documents that match certain criteria.

10.

```
db.movies.aggregate([{"$match":{"year":2007}},{"$group": {"_id": "year","averagevote":{"$avg":"$Votes"}}}])
```

```
db.movies.aggregate([{"$match":{"year":{“$eq”:2007}}},{"$group": {"_id":”$year","averagevote":{"$avg":"$Votes"}}}])
```

```
db.movies.aggregate([{"$match":{"year":2007}},{"$group":{"_id": ”$year","averagevote":{"$avg":"$Votes"}}}])
```

11. mongoexport -u root -p MTUyMjUtc21pdHNo --authenticationDatabase admin --db entertainment --collection movies --out partial_data.csv --type=csv --fields _id, title, year, rating, director

CASSANDRA

12. CREATE KEYSPACE entertainment WITH replication = { 'class' : 'SimpleStrategy' , 'replication_factor': 3};
USE entertainment;
CREATE TABLE movies (id text PRIMARY KEY, title text, year text, rating text, director text);
COPY entertainment.movies (id, title, year, rating, Director) FROM 'partial_data.csv' WITH DELIMITER = ',' AND HEADER = TRUE;

13. USE entertainment
SELECT COUNT(*) FROM movies;

14. CREATE INDEX rating_index oN movies(rating);

15. USE entertainment
SELECT COUNT(*) FROM movies WHERE rating = ‘G’;