Introduction to data services on IBM Cloud

Unit objectives

- Describe different databases types and capabilities
- Describe the main types of data services in IBM Cloud.
- Explain the benefits of IBM Cloudant.
- Access Cloudant databases and documents on IBM Cloud.
- Use HTTP APIs to interact with Cloudant database.

Introduction to databases

Topics

- Introduction to databases
- Data services in IBM Cloud
- IBM Cloudant

Importance of data

- Data is a set of facts, statistics, or figures.
- Raw data is processed to produce useful information.
- Structured data versus unstructured data.
- Leading organizations excel at capitalizing on data.

How data is stored

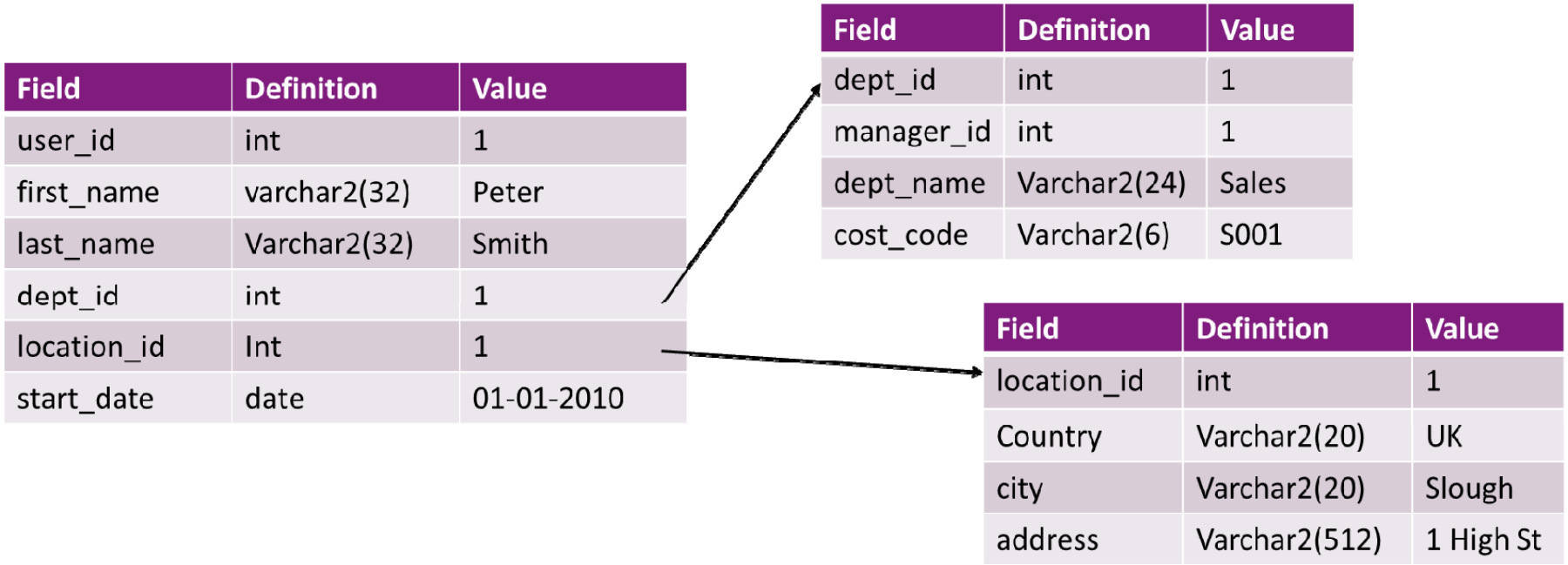
- Flat files (including XML files)
- Excel spreadsheets
- Relational databases (for example, Db2, MySQL, and PostgreSQL)
- NoSQL databases (Cloudant, MongoDB, and Redis)
- Object-based storage (IBM Cloud Object Storage)

Database and data model

- A database is a collection of information that is organized so that the data can easily be accessed, managed, and updated.
- Modern organizations use various databases types to organize and store their data.
- A data model is a conceptual representation of the data structures that are required by the database.

Relational databases

A *relational database* is a persistent storage mechanism whose data is stored in tables with a well-defined relationship between database tables.



Field	Definition	Value
user_id	int	1
first_name	varchar2(32)	Peter
last_name	Varchar2(32)	Smith
dept_id	int	1
location_id	Int	1
start_date	date	01-01-2010

Field	Definition	Value
dept_id	int	1
manager_id	int	1
dept_name	Varchar2(24)	Sales
cost_code	Varchar2(6)	S001

Field	Definition	Value
location_id	int	1
Country	Varchar2(20)	UK
city	Varchar2(20)	Slough
address	Varchar2(512)	1 High St

NoSQL databases

- A NoSQL database provides a mechanism for storage and retrieval of data that is modeled by means other than the tabular relations that are used in relational databases.
- Main key characteristics:
  - Highly scalable
  - Flexible data schema
- Different types of NoSQL databases:
  - Key-Value
  - Document
  - Columnar
  - Graph

Different types of NoSQL databases: Key-value

- A simplistic data schema with a simple list of keys and values. The key is a pointer to the value.
- The key can be a hash value or a real value, such as an email address or other unique reference number. The contents of the value are not formatted.
- Key-value stores allow fast access.

Key	Value
Name	Peter Smith
Department	Sales
Location	{country: ‘UK’, city: ‘Slough’}

Different types of NoSQL databases: Document

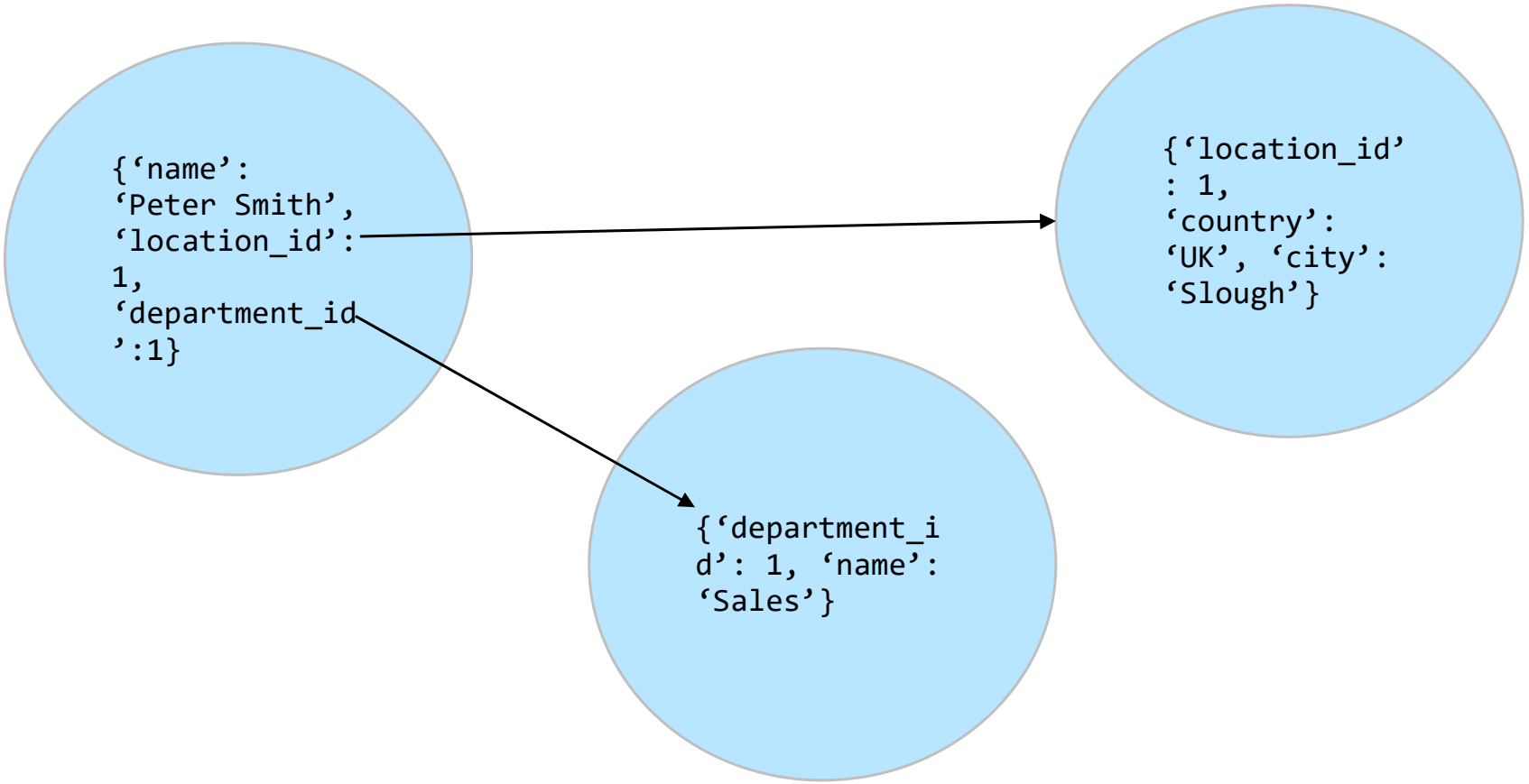
Document NoSQL databases pair each key with a complex data structure that is known as a *document*. Documents can contain many different key-value pairs, key-array pairs, or nested documents.

Different types of NoSQL databases: Columnar

- A *columnar* database is a database that stores data in columns instead of rows. For example:
  - ID: 1, Name: Ahmed, Age: 29, Weight: 65
  - ID: 2, Name: Ben, Age: 34, Weight: 70
  - ID: 3, Name: John, Age: 32, Weight: 73
- Each column is stored in the following database records:
  - Ahmed: 1, Ben: 2, John: 3
  - 29:1, 34:2, 32:3
  - 65:1, 70:2, 73:3

Different types of NoSQL databases: Graph

- Graph-based databases can process complex queries more easily than the relational data model because they use an intuitive data model with a simple modeling of the nodes and the relationship as the edges.
- Queries generally use the *graph traversal process* or algorithms



Data services in IBM Cloud

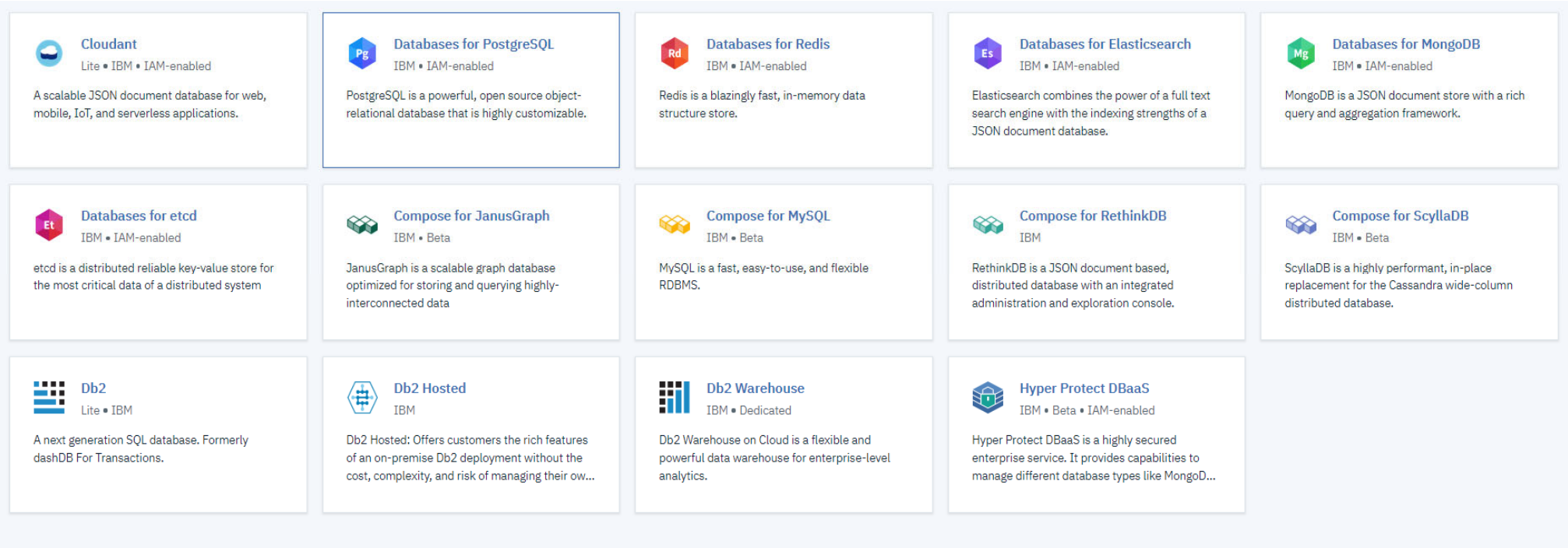
Topics

- Introduction to databases
- Data services in IBM Cloud
- IBM Cloudant

Data services in the IBM Cloud catalog

- Databases services
- Storage services
- Analytics services

Databases services



Storage services

Block Storage
IBM

Persistent iSCSI based storage with high-powered performance and capacity up to 12TB.

Db2 Warehouse
IBM • Dedicated

Db2 Warehouse on Cloud is a flexible and powerful data warehouse for enterprise-level analytics.

File Storage
IBM

Fast and flexible NFS-based file storage with capacity options from 20GB to 12TB.

IBM Cloud Backup
IBM

A fast and flexible backup solution that is managed by IBM Cloud and provides capacity options to scale perfectly with your needs.

Object Storage
Lite • IBM • IAM-enabled

Provides flexible, cost-effective, and scalable cloud storage for unstructured data.

Actifio GO
Third Party • IAM-enabled

ActifioGo in the IBM Cloud: Backup direct to cloud for VMware virtual machines

box
Third Party

Powering Content and data for your application. Whether you are building a line of business app, content management software or need to display...

Analytics services

Analytics Engine
Lite • IBM • IAM-enabled

Flexible framework to deploy Hadoop and Spark analytics applications.

BigInsights for Apache Hadoop (Subscription)
IBM • Deprecated

Provision managed bare metal Apache Hadoop clusters for production use or POCs at scale.

Decision Optimization
IBM • Deprecated

Develop optimization applications, such as planning or scheduling, using our APIs to connect to the CPLEX optimization engines.

Geospatial Analytics
IBM

Expand the boundaries of your application. Leverage real-time geospatial analytics to track when devices enter, leave or hang out in defined...

IBM Cognos Dashboard Embedded
Lite • IBM • IAM-enabled

Bring data to life directly from your application with this powerful and easy-to-use visualization service.

Master Data Management
IBM

IBM® Master Data Management (MDM) on Cloud helps businesses gain a trusted view of data in a hybrid computing environment.

SQL Query
Lite • IBM • IAM-enabled

Read, analyze, and store data in Cloud Object Storage with ANSI SQL.

Streaming Analytics
Lite • IBM • IAM-enabled

Leverage IBM Streams to ingest, analyze, monitor, and correlate data as it arrives from real-time data sources. View information and events as...

Weather Company Data
IBM

Use the Weather Company Data for IBM Cloud service to incorporate weather data into your IBM Cloud applications.

AccountScore
Third Party

AccountScore Open Banking & transaction analytics

Other related data services

Blockchain Platform 2.0
IBM • IAM-enabled

Try the next generation of the IBM Blockchain Platform for free, with all the tooling you need to deploy, manage, and govern blockchain networks.

Blockchain
IBM

IBM Blockchain Platform is a flexible software-as-a-service offering that simplifies the blockchain journey of developing, governing, and operating ...

Messages for RabbitMQ
IBM • IAM-enabled

RabbitMQ is an open source multi-protocol messaging broker.

Compose Enterprise
IBM

IBM Compose Enterprise is a service which provides a private isolated cluster for IBM Cloud users to optionally provision their Compose...

Db2
Lite • IBM

A next generation SQL database. Formerly dashDB For Transactions.

Db2 Warehouse
IBM • Dedicated

Db2 Warehouse on Cloud is a flexible and powerful data warehouse for enterprise-level analytics.

Hyper Protect DBaaS
IBM • Beta • IAM-enabled

Hyper Protect DBaaS is a highly secured enterprise service. It provides capabilities to manage different database types like MongoDB ...

Informix
IBM

IBM Informix on Cloud helps businesses gain a trusted view of data in a hybrid computing environment.

SQL Query
Lite • IBM • IAM-enabled

Read, analyze, and store data in Cloud Object Storage with ANSI SQL.

GEO Web Services
Third Party

Adding geo-intelligence to your business.

InfluxCloud
Third Party

A modern time series data platform for metrics & events

IBM Cloudant

Topics

- Introduction to databases
- Data services in IBM Cloud
- IBM Cloudant

Cloudant capabilities

- Database as a service (DBaaS): Provision and scale according to your requirements.
- Data is stored as documents in JSON format: Schema-less NoSQL format.
- Simple API: REST-based.
- Cloudant search.
- IBM Cloudant Geo.
- Offline First mobile web apps capabilities.
- Synchronization feature for disconnected Android and Apple apps.
- Client libraries for developing your own application.

Benefits of IBM Cloudant

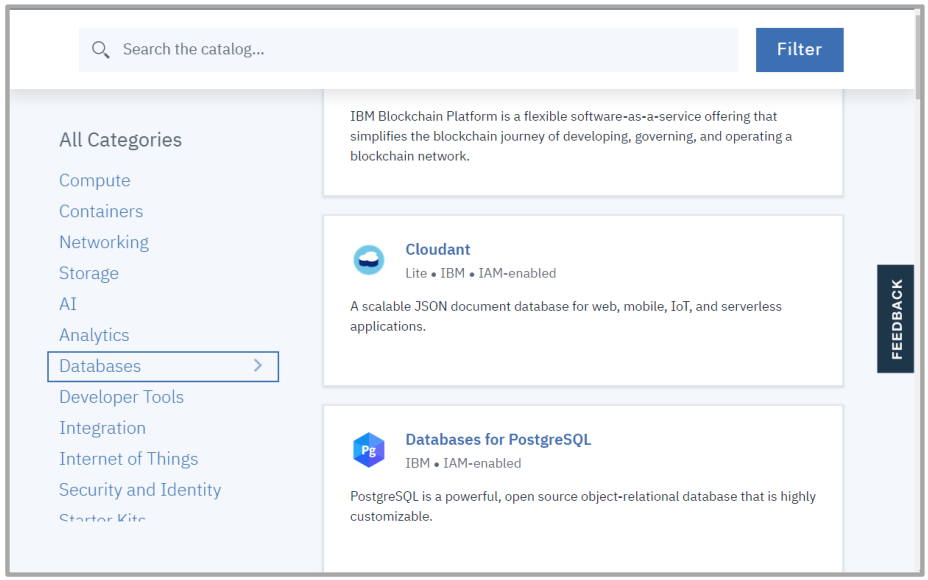
- High scalability
- High availability, including data replication worldwide
- Satisfied by eventually consistent results (”stale” reads are better than no reads)
- High performance at large (> 1 TB) scale
- ACID transactions at the document level
- Fully managed database as a service
- Powerful serverless API

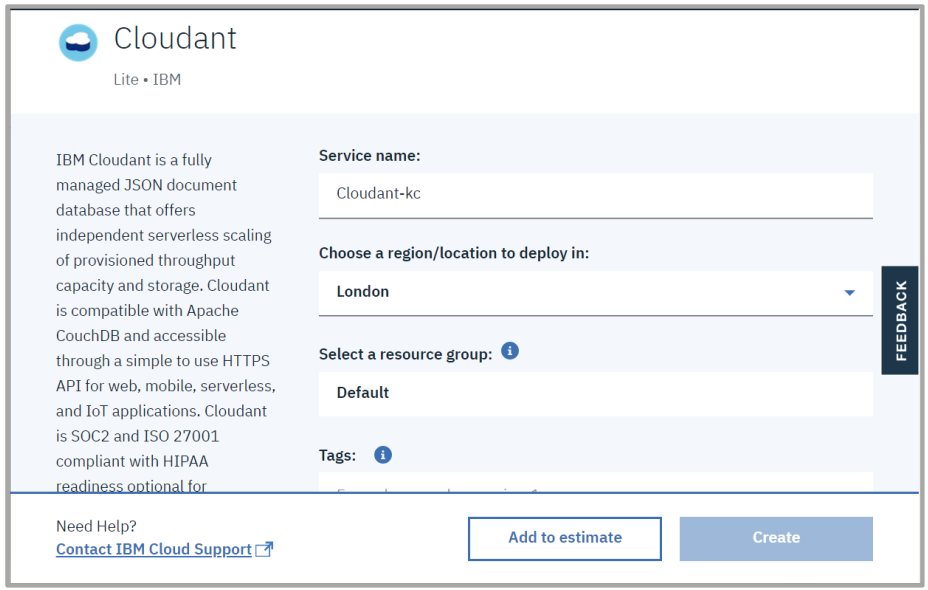
Documents in Cloudant

- Documents are JSON objects.
- Cloudant documents are containers for the data.
- All documents have the following unique mandatory fields:
  - Unique `_id`
  - `_rev`
- In addition to the two mandatory fields, documents can contain any other content that is expressed in the JSON format.

Getting started with Cloudant in IBM Cloud

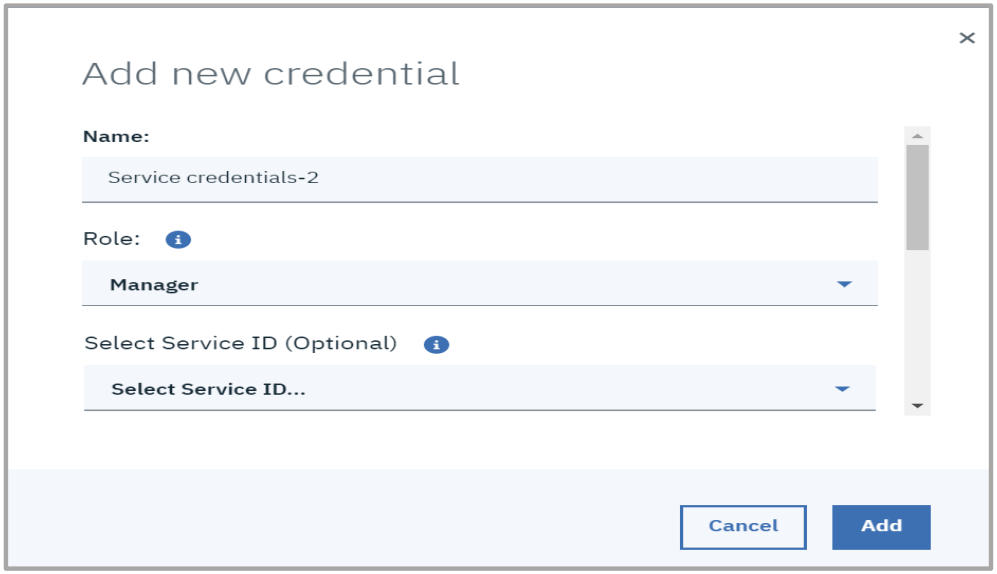
1. From the IBM Cloud Dashboard, click **Create resource**.
2. In the Catalog page, select **Databases** under categories and then select **Cloudant**.
3. Enter a descriptive name in the **Service name** field.
4. Select a **region**, **resource group**, and **authentication method**.
5. Select a **pricing plan** that fits your needs. You can always start with the free plan and upgrade later through the Cloudant Dashboard.
6. Click **Create**.

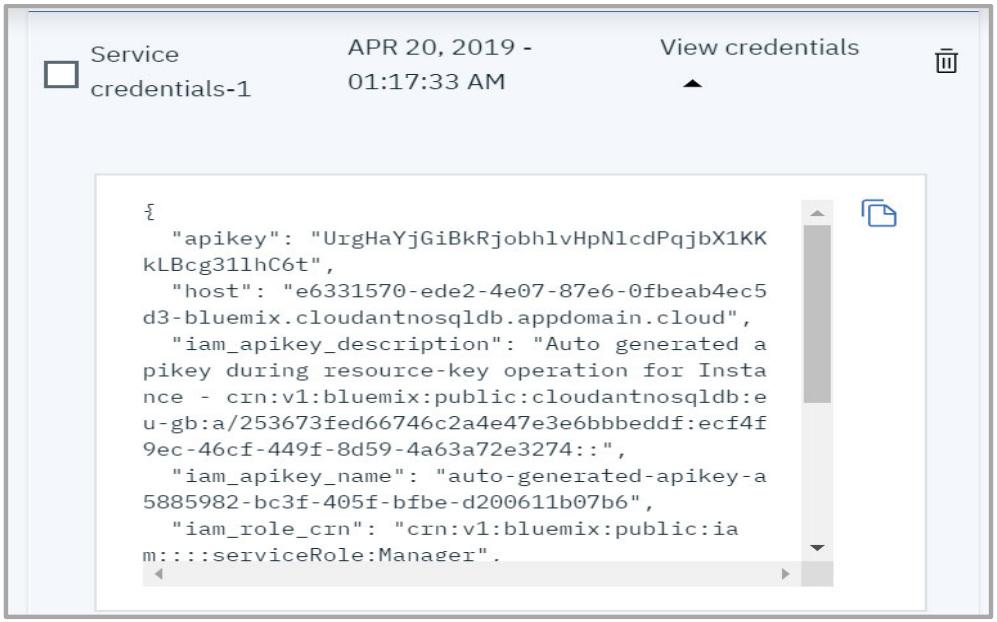




Cloudant with IBM Cloud: Creating credentials

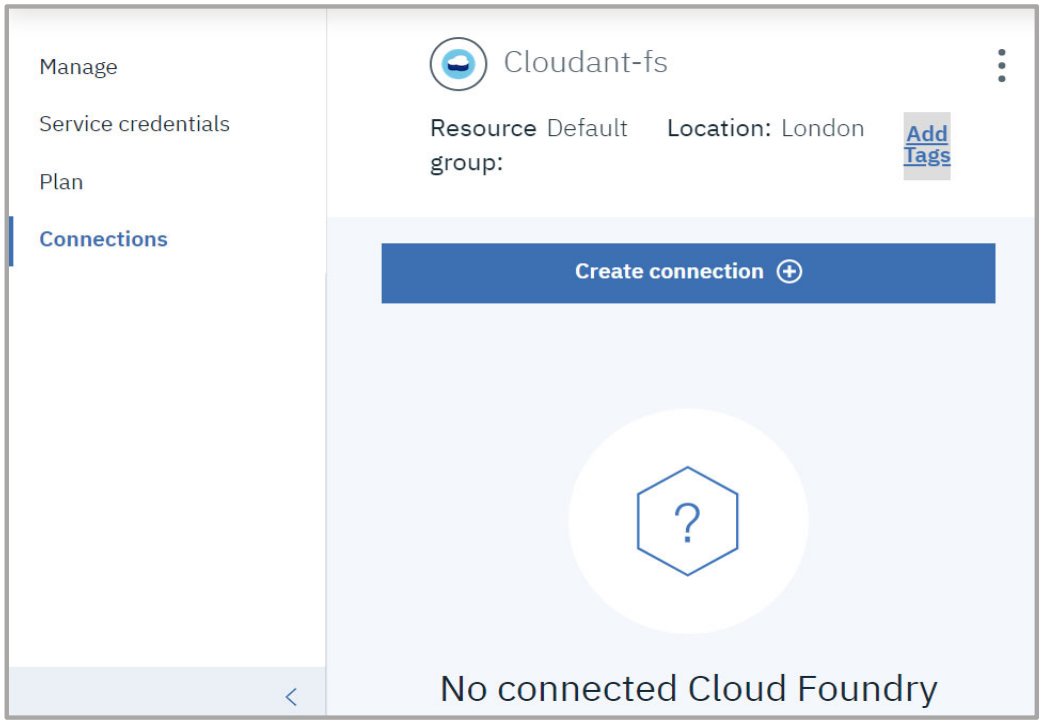
1. From the resource list, select the Cloudant service instance to open it.
2. Select the **Credentials** tab, and click **New Credential +**.
3. From the Add New Credential dialog, provide a **Name**, **Role**, **Service ID (optional), and Inline Configuration Parameters (Optional)**.
4. Click **Add** to generate the new service credential.



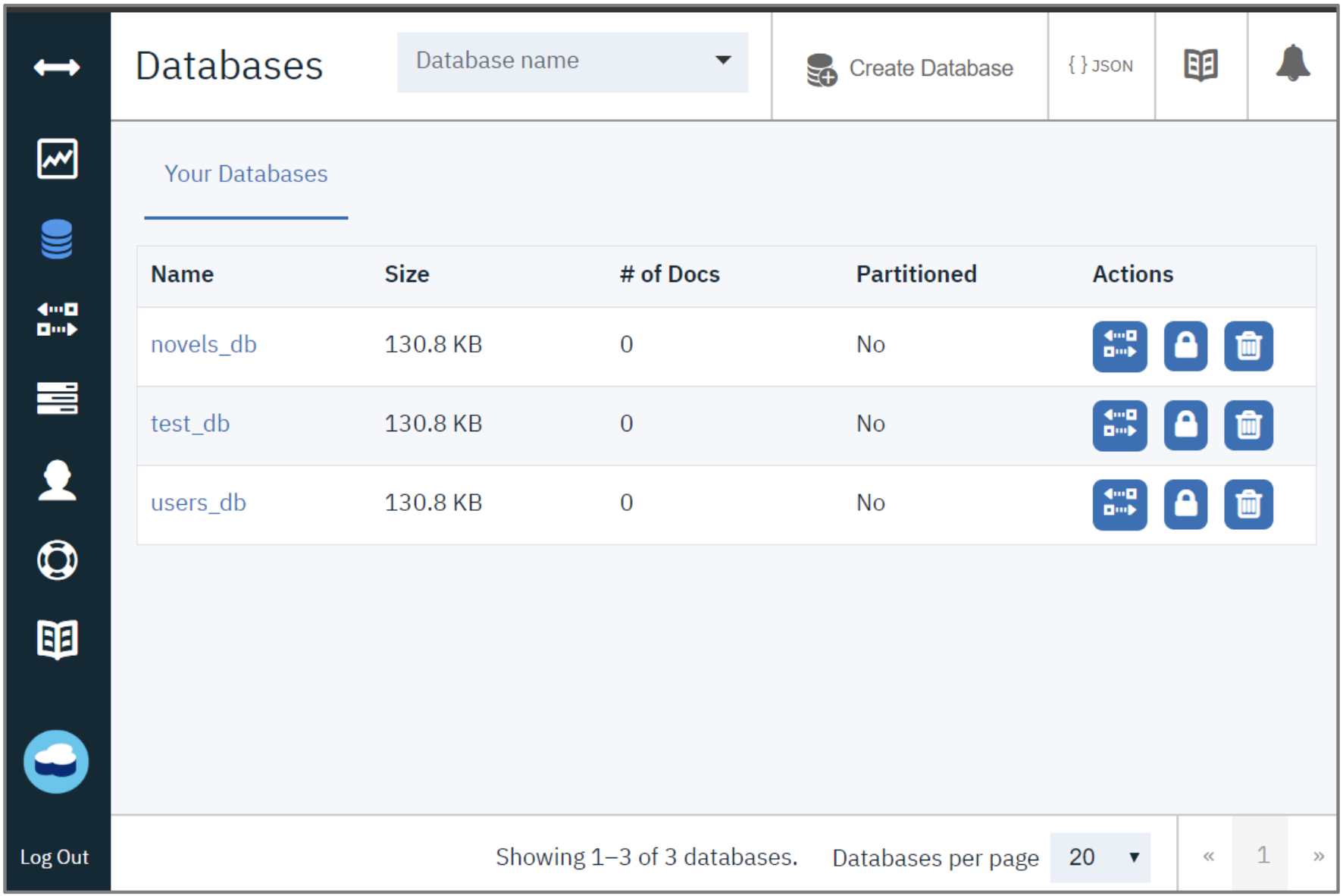


Cloudant with IBM Cloud: Creating connections

1. From the resource list, select the Cloudant service instance to open it.
2. Select the **Connections** tab, and click **Create connection**.
3. Click **Connect** for the row of the app for which you want to create the connection.
4. Select an access **Role**, **Service ID** and optionally add configuration parameters in JSON format.
5. Click **Connect & restage app**.



Cloudant Dashboard



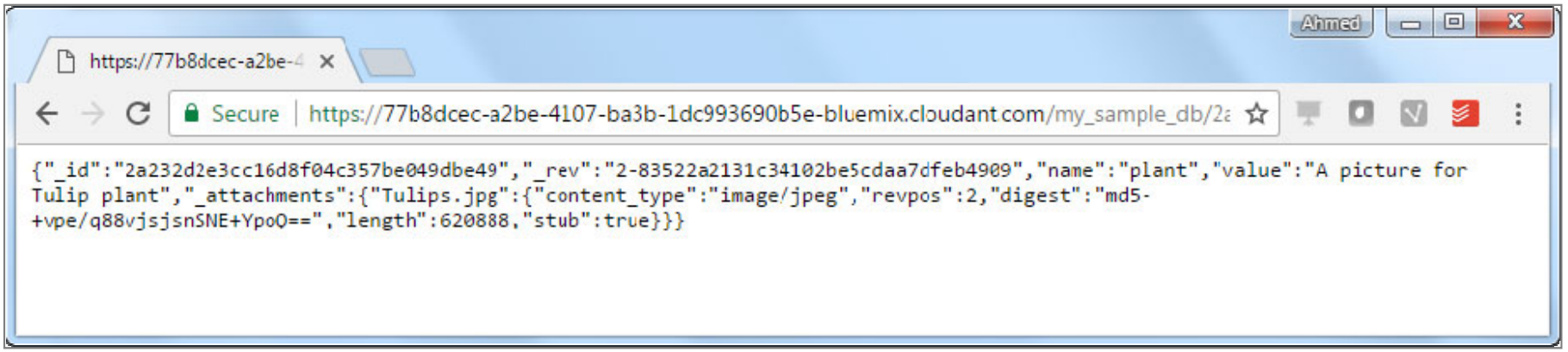
Cloudant HTTP API

- Simple, web-based access to Cloudant data:
  - HTTP API.
  - Includes wrappers for various languages, such as Java and JavaScript.
  - Every document in the DB is accessible as JSON by using a URL.
- HTTP request methods include:
  - `GET`.
  - `PUT`.
  - `POST`.
  - `DELETE`.

Reading a document in Cloudant

To access a document with the Cloudant API, issue a **GET** request to the following URL:

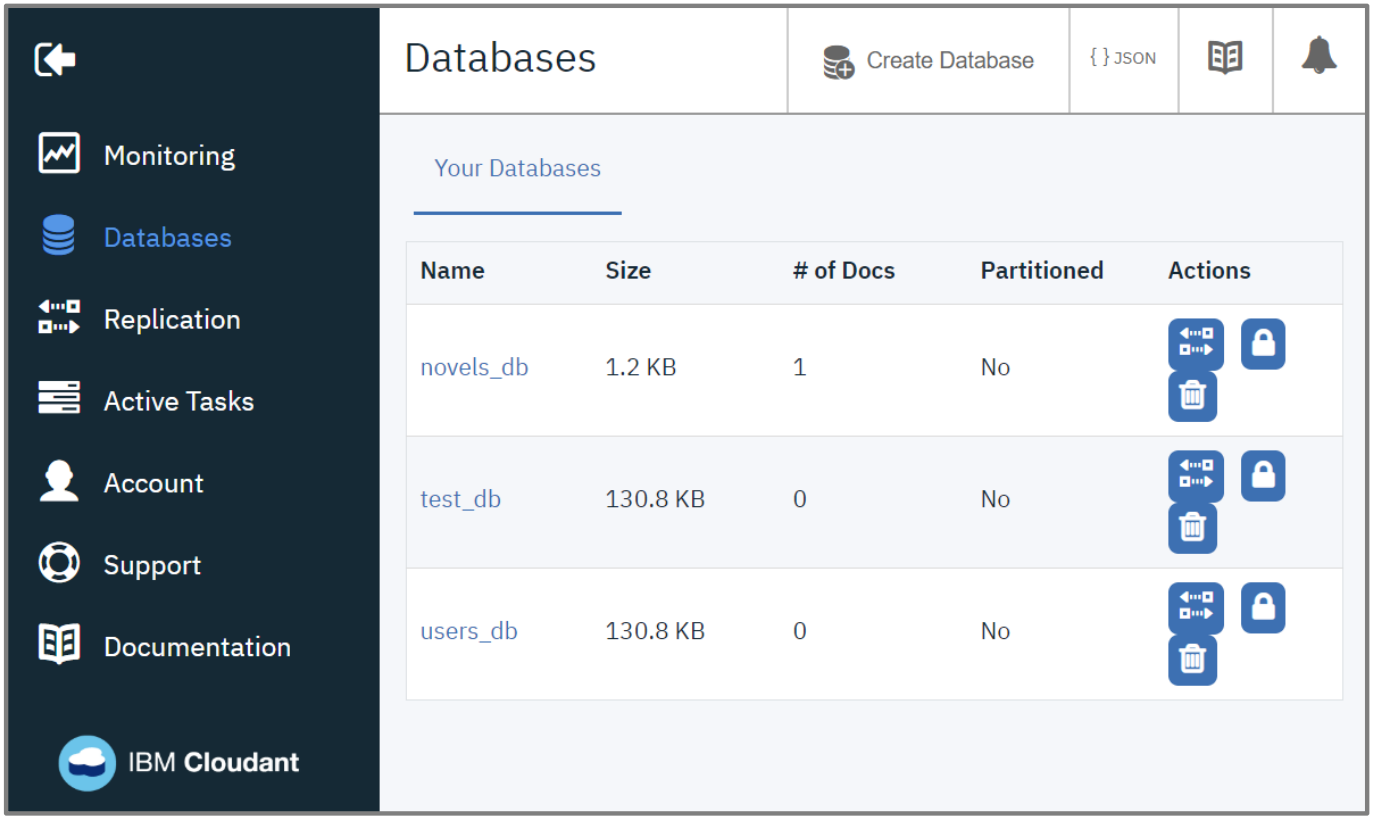
`https://$USERNAME.cloudant.com/$DATABASE/$DOCUMENT_ID`



View all documents

To view all documents at a database, issue a **GET** request to the following URL:

`https://$USERNAME.cloudant.com/$DATABASE/_all_docs?include_docs=true`



More Cloudant HTTP APIs

- Create a document:

  POST https://$USERNAME.cloudant.com/$DATABASE
  with the document's JSON content in the body.

- Update a document:

  PUT https://$USERNAME.cloudant.com/$DATABASE/$DOCUMENT_ID
  with the updated document JSON content, including latest _rev in the body.

- Delete a document:

  DELETE
  https://$USERNAME.cloudant.com/$DATABASE/$DOCUMENT_ID?rev=$REV

Cloudant Query

- Cloudant Query is a declarative JSON querying syntax for Cloudant databases.
- To query a document, issue a `POST` request to https://$USERNAME.cloudant.com/$DATABASE/_find with a selector in the body.
- A *selector* is a JSON object describing the criteria that is used to select documents.
- Example of a Cloudant query body:

```
{
  "selector": {
    "lastname": "Brown",
    "location": "New York City, NY"
  },
  "fields": [
    "firstname",
    "lastname",
    "location"
  ]
}
```

Cloudant indexes

- Indexes enable quick access to a portion of the data.
- To create an index, issue a `POST` request to https://$USERNAME.cloudant.com/$DATABASE/_index with a body that contains index field names, an index name, and an index type.
- Example of creating an index body:

```
{
      "index": {
          "fields": ["foo"]
      },
      "name": "Movie_name-text",
      "type": "text"
  }
```

HTTP status codes

- Status and errors in Cloudant are reported by using a combination of the following data:
  - HTTP status code
  - Corresponding data in the body of the response data
- Example status codes:
  - `200 - OK`
  - `201 - Created`
  - `400 - Bad request`
  - `401 - Unauthorized`
  - `404 - Not Found`
- Example detail that is supplied in JSON format, following a 404 status code:

```
{
  "error": "not_found",
  "reason":"missing"
}
```

Unit summary

- Describe different databases types and capabilities
- Describe the main types of data services in IBM Cloud.
- Explain the benefits of IBM Cloudant.
- Access Cloudant databases and documents on IBM Cloud.
- Use HTTP APIs to interact with Cloudant database.

Exercise 3: IBM Cloud with Cloudant

Exercise objectives

- This exercise demonstrates how you can create a Cloudant database service on IBM Cloud.
- After completing this exercise, you should be able to perform the following tasks:
  - Create an instance of the Cloudant service on IBM Cloud.
  - Create service credentials by using IBM Cloud Identity and Access Management (IAM).
  - Access the Cloudant documentation.
  - Explore the features of the Cloudant Dashboard.
  - Create, read, update, and delete Cloudant documents by using HTTP APIs.
  - Verify the data that is stored in the database from the Cloudant Dashboard.
  - Create indexes and query Cloudant documents by using HTTP APIs.

Documentation and other information sources

- Cloudant documentation:

  https://cloud.ibm.com/docs/services/Cloudant?topic=cloudant-getting-started

- IBM Cloud Data Services documentation:

  https://developer.ibm.com/clouddataservices/

- IBM Cloudant videos on YouTube:

  https://www.youtube.com/channel/UCSMx6Fgq1RJLq58em2mJeKQ

- IBM Developer articles and resources:

  https://www.ibm.com/developerworks

- Online Learning Labs:

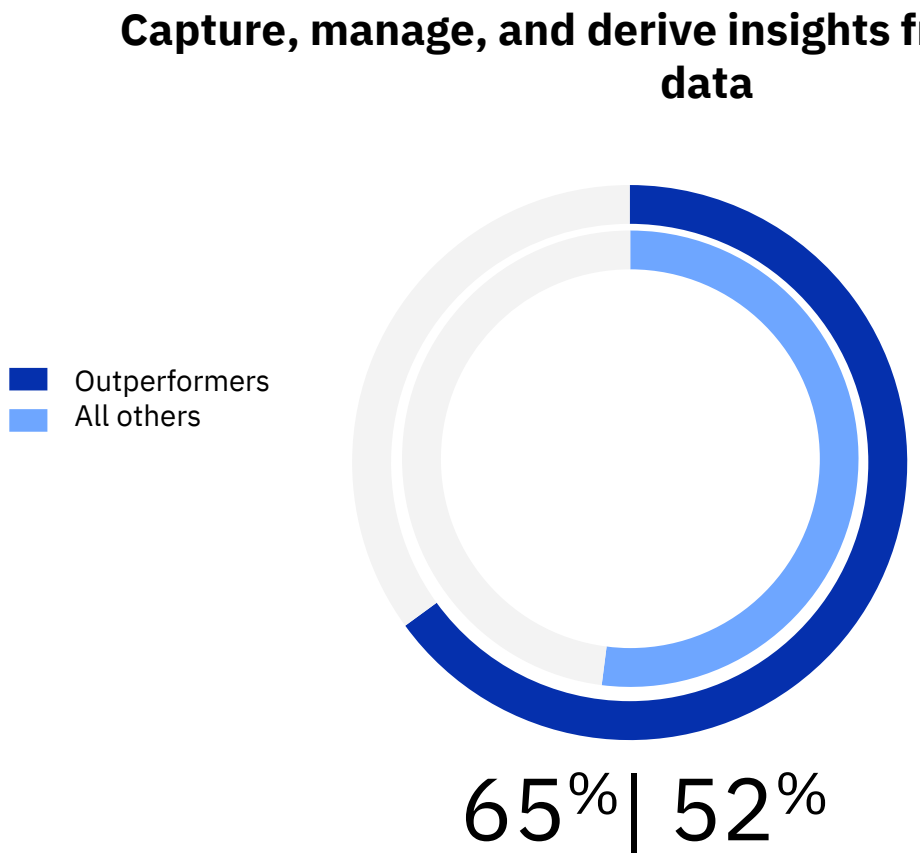
  https://www.ibm.com/cloud/garage/category/courses

- Get IBM Cloud Essentials Open Badge:

  https://developer.ibm.com/courses/category/databases/

Importance of data (backup slide)

Leading organizations excel at capitalizing on data.



- Source: 2018 AI Survey: A8.5: For the following statement, indicate the level of rigor with which your organization performs the following activities - Capture, manage, and access business, technology, and operational information on key corporate data, N= 4992

Atomicity – Consistency – Isolation – Durability (backup slide)

- **Atomicity:** Either all tasks in a transaction are performed or none of them are. If one element of a transaction fails, the entire transaction fails.
- **Consistency:** The transaction does not violate any protocols or rules that are defined in the system and the database must remain in a consistent state at the beginning and end of a transaction; there are never any half-completed transactions.
- **Isolation:** No transaction has access to any other transaction that is in an intermediate or unfinished state. This is required for both performance and consistency of transactions within a database.
- **Durability:** After the transaction is complete, it persists as complete and cannot be undone; it survives system failure, power loss, and other types of system breakdowns.

Basically Available – Soft State – Eventual consistency (backup slide)

- **Basically Available:** There is a response to any request, *but* that response might still "fail" to obtain the requested data or the data might be in an inconsistent or changing state, much like waiting for a check to clear in your bank account.
- **Soft state:** The state of the system can change over time, so even during times without input there might be changes going on due to *eventual consistency*. Thus, the state of the system is always *soft*.
- **Eventual consistency:** The system *eventually* becomes consistent after it stops receiving input. The data propagates to everywhere it should sooner or later, but the system continues to receive input and is not checking the consistency of every transaction before it moves onto the next one.

Features of NoSQL databases (backup slide)

- Key characteristics of NoSQL technologies:
  - Highly scalable.
  - Flexible data schema.
- Why is it more flexible and scalable?
  - It does not require a predefined data model for storage, such as specific row and column names and sizes.
  - It is optimized to work on distributed hardware.
  - It uses relatively simple queries that can be processed quickly across much larger data sets.
- Therefore, these databases are well-suited to applications that are characterized by:
  - Large amounts of data.
  - Low latency requirements.
  - Non-relational data.
  - Unstructured data.
  - Simple data queries (does not require multistep transactions).
- Follow the BASE design properties.

Cloudant Index and other types of queries (backup slide)

CRUD – Document	Primary Index	Secondary Index (view)	Search Index	Geospatial Index	Cloudant Query
					
• Direct document lookup by _id	• Exists "OOTB" • Stored in a b-tree • Primary key > doc._id	• Built by using MapReduce • Stored in a b-tree • Key > user-defined fields	• Built by using Lucene • FTI: Any or all fields can be indexed	• Stored in R* tree • Lat/Long coordinates in GeoJSON	• "Mongo-style" querying • Built natively in erlang
• Use when you want a single document and can find by its _id	• Use when you can find documents based on their _id • Pull back a range of keys	• Use when you need to analyze data or get a range of keys • Examples: count data fields, sum/average numeric results, advanced stats, group by date, and so on.	• Ad hoc queries • Find documents based on their contents • Can do groups, facets, and basic geo queries (bbox and sort by distance)	• Complex geometries (polygon, circularstring, and so on) • Advanced relations (intersect, overlaps, and so on)	• Ad hoc queries • Many operators (>, <, IN, OR, AND, and so on) • Intuitive for people who come from Mongo or SQL backgrounds

Cloudant best practices (backup slide)

- Performance considerations.
- Large attachments.
- Replication with `_replicate` endpoint does not persist.
- Use show and list functions sparingly.
- Data design considerations:
  - Eventually Consistent system.
  - Doc updates and immutable data.
  - Balancing the following needs:
    - Denormalizing data to minimize HTTP requests
    - Using fine-grained documents to avoid conflicts
  - Migrating from relational or SQL to Cloudant.
  - Simulating transactions and ACID compliance.
  - MVCC is not a version control system.
  - Organizing docs into databases.