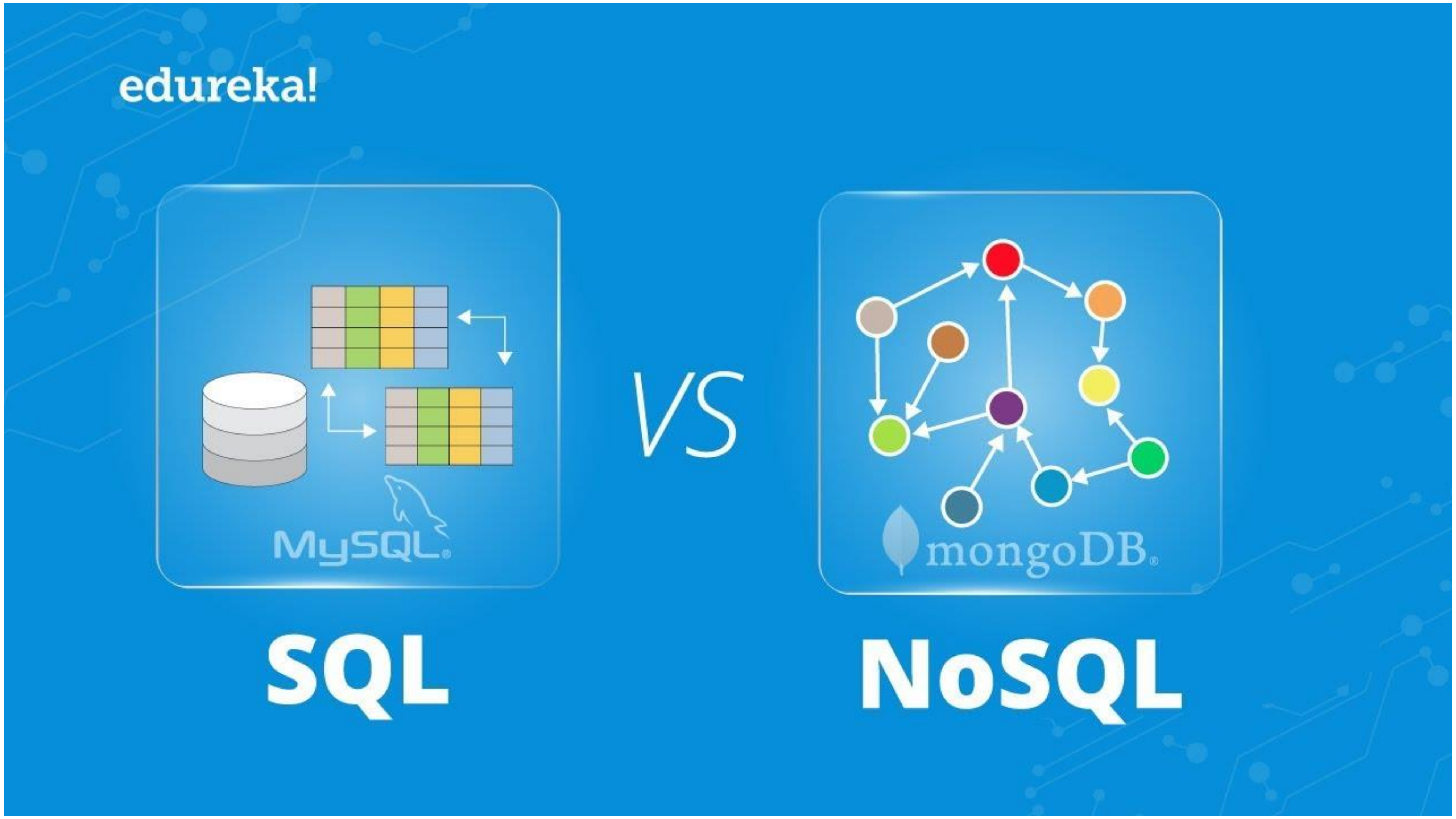
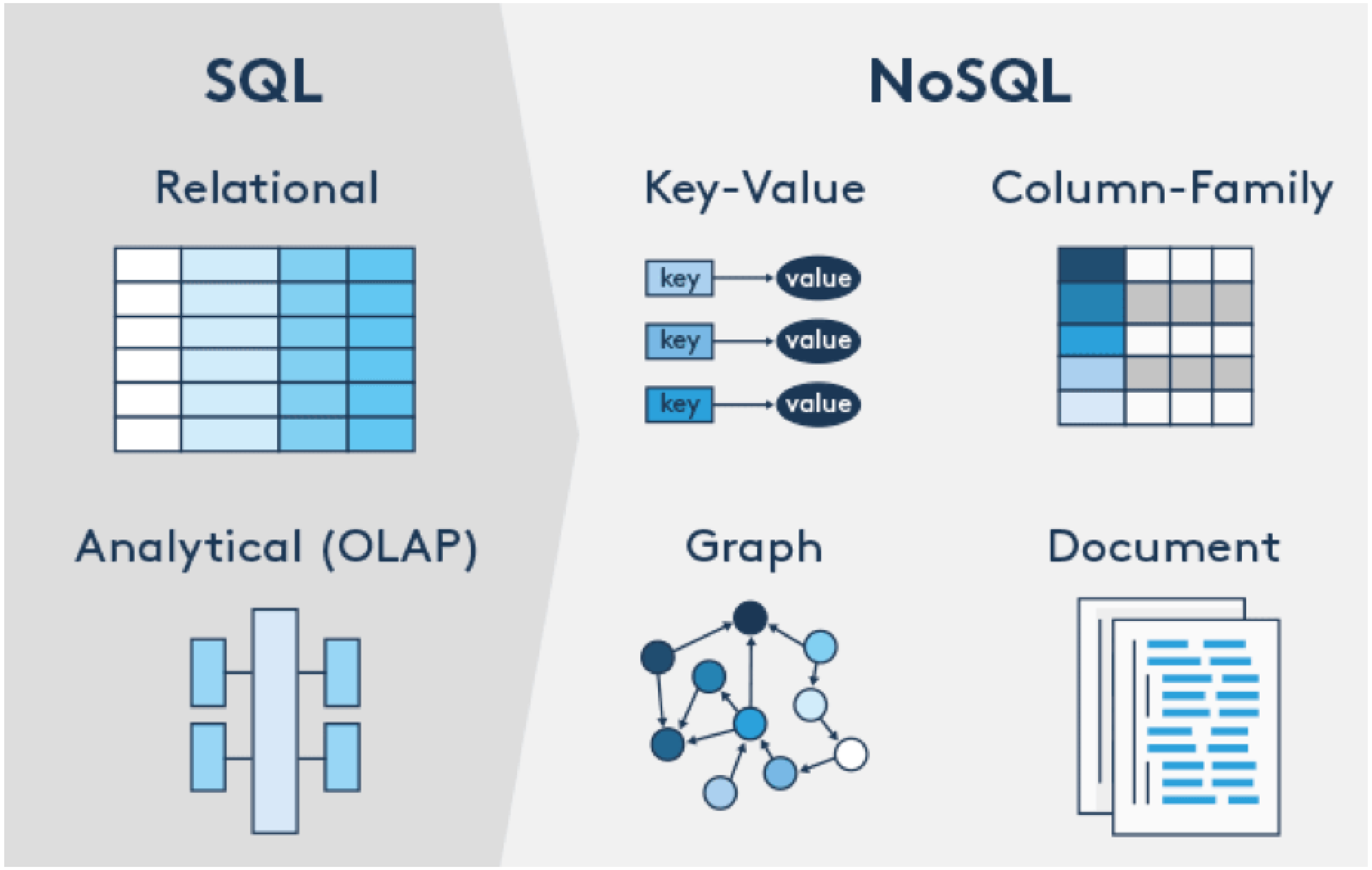
SQL vs NOSQL


edureka!
MySQL
VS
mongoDB
SQL
NoSQL


SQL
NoSQL
Relational
Key-Value
Column-Family
key
value
key
value
key
value
Analytical (OLAP)
Graph
Document

SQL vs NoSQL CRUD Syntax

C => Create (INSERT INTO)

R => Read (SELECT)

U => Update

D => Delete

- NoSQL ("non SQL" or "not only SQL") databases store data in a format other than relational tables.
- NoSQL databases come in a variety of types characterized by their data model.
- Examples include **document**, **key-value**, **wide-column**, and **graph**.
- They typically provide flexible schemas and the ability to easily scale with large amounts of data and high user loads.

A. Return all records from table

SQL	NoSQL

Select * from table_Name
 	
EX:
Select * from book;
 	
return all columns
	db.collection_name.find();
	
	EX:
	db.book.find();
	
	return all fields

Select col1,col2,col3,… from table_name
 	
EX:
Select title, author, price from book;
 	
Return the Specified Fields
	db.book.find({field1: 1, field2: 1, field3: 1, field4: 0});
	
	Ex:
	db.book.find({title: 1, author: 1, price: 1, _id: 0});
	db.book.find({title: true, author: true, price: true, _id: false});
	
	Return the Specified Fields and excluded the _id Field Only

B. return specific columns where condition

SQL	NoSQL

Select col_name from table_name where condition
EX:1
SELECT title FROM book WHERE price > 10;
EX:2
SELECT title FROM book WHERE price < 50;
EX:3
SELECT * FROM book WHERE price >= 30 and price <= 70;
EX:4
SELECT title, author, date
FROM book
WHERE date BETWEEN '1-june-1992' AND '15-december-1993'
	db.collection_name.find(
	{ field1: { $gt: 10 } }, { _id: 0, title: 1 });
	$gt means greater than
	$lt means less than
	EX1:
	db.book.find(
	{ price: {$gt: 10 } }, { _id: 0, title: 1 });
	EX2:
	db.book.find(
	{ price: {$lt: 50 } }, { _id: 0, title: 1 });
	EX3:
	db.book.find(
	{date:{$gt:'1-june-1992', $lt:'15-december-1993’}});

C. count the number of SitePoint books

SQL	NoSQL

SELECT COUNT (col_name) FROM table_name
WHERE condition;
 	
EX:
SELECT COUNT (author) FROM book
WHERE author= 'SitePoint';
	db.collection_name.count({
	  "field_name": "value_condition"
	});
	
	EX:
	db.book.count(
	{ “author": "SitePoint“ });
	
	This presumes denormalized documents are used.

D. return the number of book format types

SQL	NoSQL

SELECT column_name1, COUNT (column_name) AS \`alis\` FROM table_name GROUP BY column_name;
 	
\`alis\` اسم بديل
 	
EX:
SELECT format, COUNT (author) AS ‘total’ FROM book GROUP BY format;
	db.collection_name.aggregate([
	{$group:{ _id: "$format", ‘alis’: {$sum: author}}}
	]);
	
	\`alis\` اسم بديل
	
	EX:
	db.book.aggregate([
	{$group:{ _id: "$format", total: {$sum: author}}}
	]);
	
	This is known as aggregation: a new set of documents is computed from an original set.

E. insert a new book record

SQL	NoSQL
INSERT INTO table_name (col1, col2, col3, ...)	db.collection_name.insert({
VALUES (value1, value2, value3, ...);	field1: "value",field2: " value",field3: " value“ });
EX:	Ex:
INSERT INTO book (	db.book.insert({
`ISBN`, `title`, `author`	ISBN: "9780992461256",
)	title: "Full Stack JavaScript",
VALUES (	author: "Colin Ihrig & Adam Bretz"
'9780992461256',	});
'Full Stack JavaScript',	
'Colin Ihrig & Adam Bretz'	
);	

E. update a book record

SQL	NoSQL

UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;
 	
EX:
UPDATE book
SET price = 19.99
WHERE ISBN = '9780992461256’;
	db.collection.update(query, update, options)
	
	db.collection.update(
	  { condition_field: value },
	  { $set: { field: value } }
	);
	
	EX:
	db.book.update(
	  { ISBN: '9780992461256' },
	  { $set: { price: 19.99 } }
	);

F. delete all SitePoint books

SQL	NoSQL

DELETE FROM table_name WHERE condition;
 	
EX:
DELETE FROM book
WHERE author='Alfreds Futterkiste';
	db.collection.deleteOne();
	
	EX:
	db.book.deleteOne({ status: "D" });
	
	EX:
	db.book.deleteOne(
	{ "_id" : _Id("563237a41a4d68582c2509da")}
	);

G. Drop Database

SQL	NoSQL

DROP DATABASE databasename;
 	
Ex:
DROP DATABASE book;
	db.collection_Name.drop();
	
	EX:
	db.book.drop();