**Key-Value**

* Datele sunt stocate cu o key si o value caracteristica,adica un HashMap

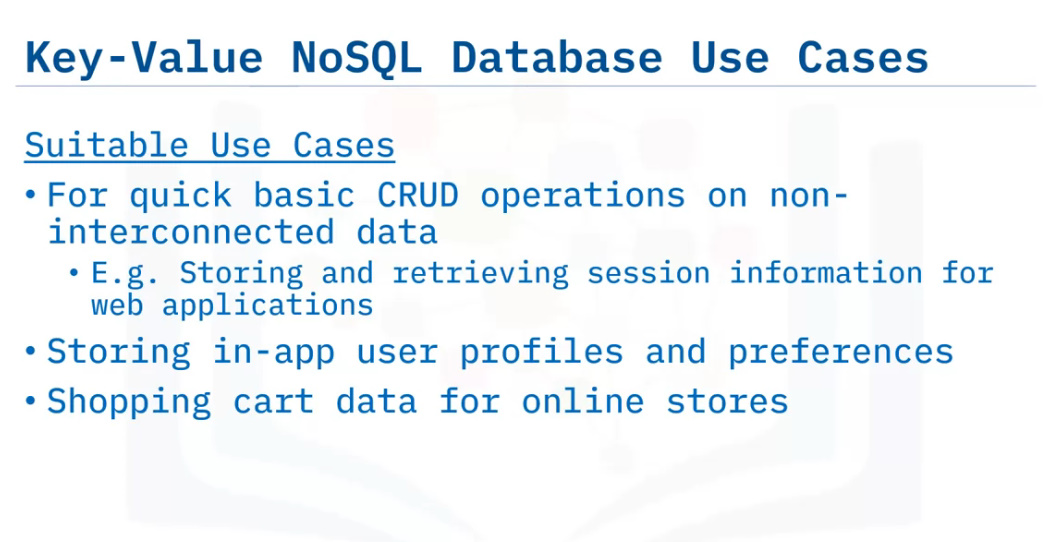
**Avantaje**

* Sunt cele mai usoare din NoSQL
* Sunt bune pentru operatii de baza din CRUD
* Sunt foarte usor impartasite in diferite noduri

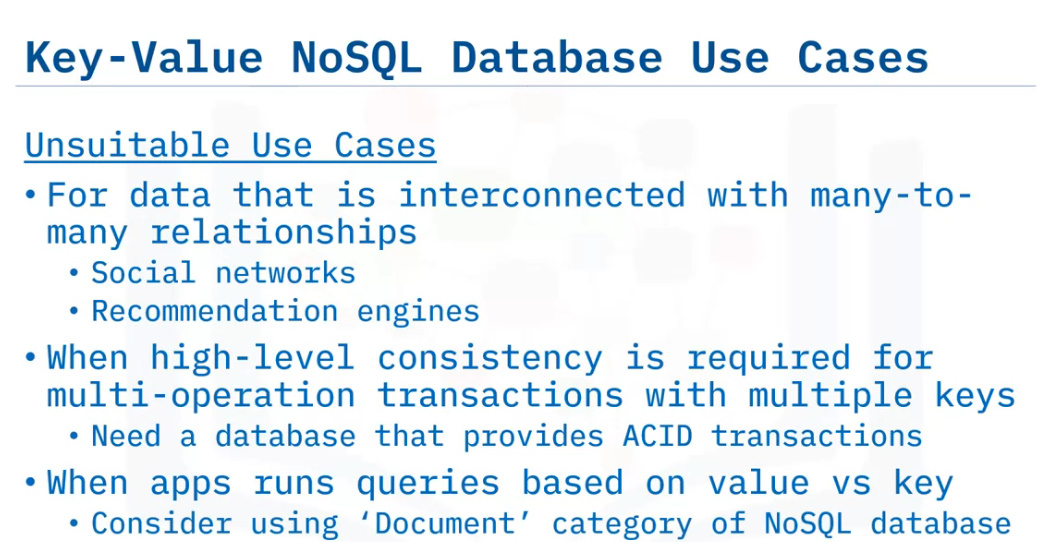
**Dezavantaje**

* Nu sunt pentru complex queries, ca joins etc.
* Atomice doar pentru single key operations
* Nu sunt bine indexate si supuse la query din cauza ca value blobs sunt opace la baza de date

**Utilizare**



**Cazuri cand nu e buna**



**DBMS**

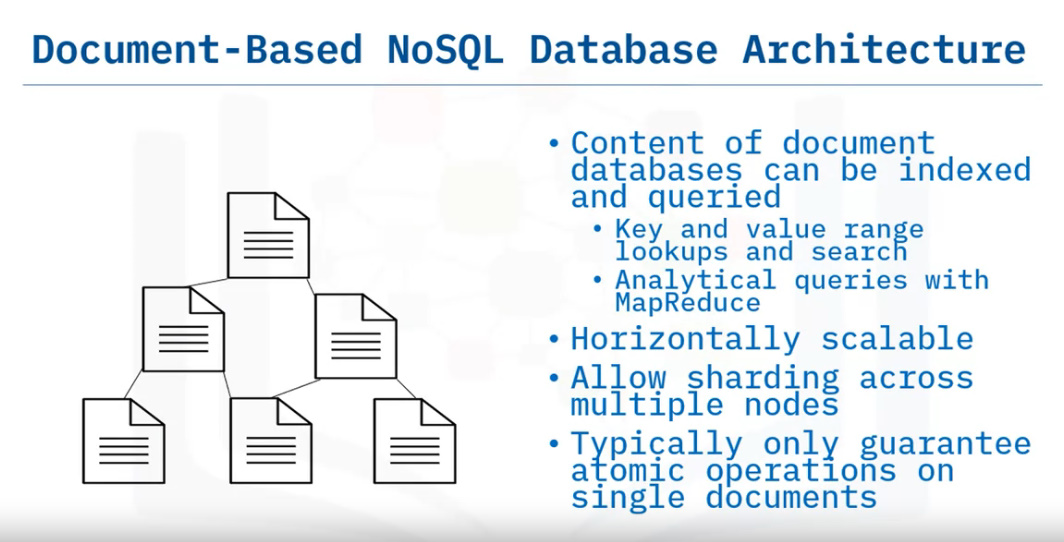


**Document**

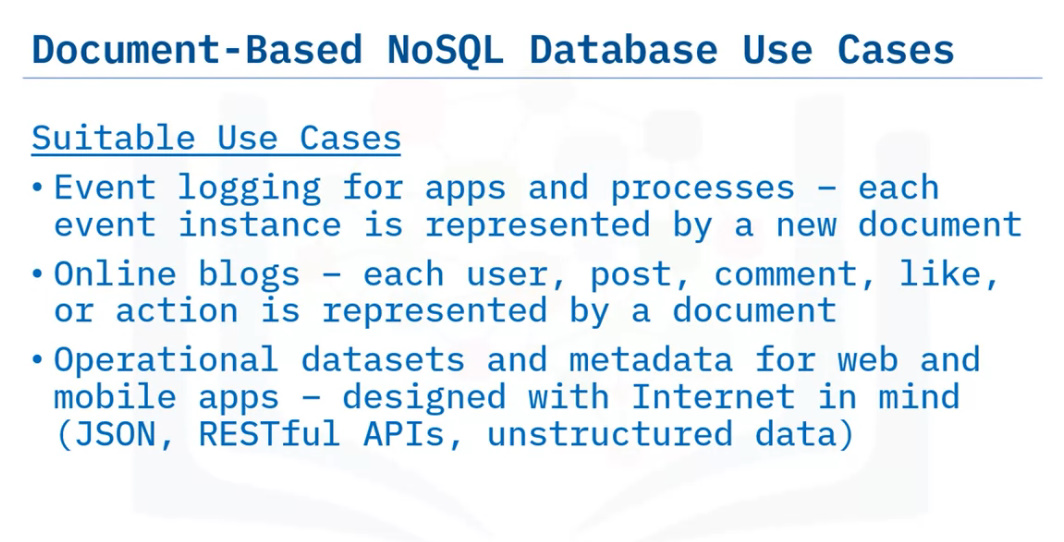
* Se bazeaza tot pe Key-Value model, doar ca face ca values sa fie vizibile si se poata lucra cu ele, si sunt stocate in documente separate.
* Sunt cele mai populare acum

**Avantaje**

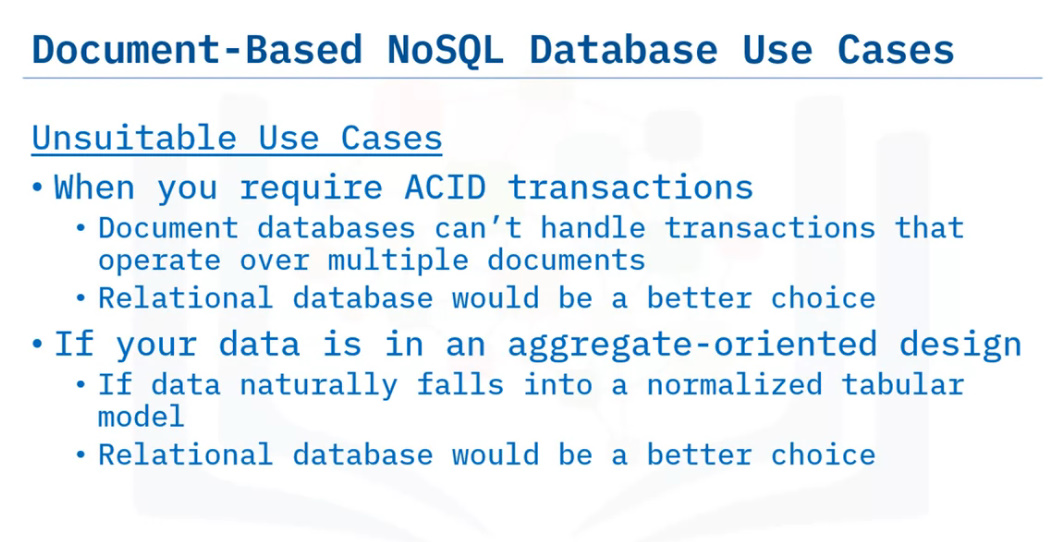
* Deoarece values sunt vizibile, pot fi efectuat query asupra lor
* Fiecare bucata de data este un document
* Este stocat in JSON sau XML
* Fiecare document ofera o schema flexibila, si poate contine date aranjate diferit de altele



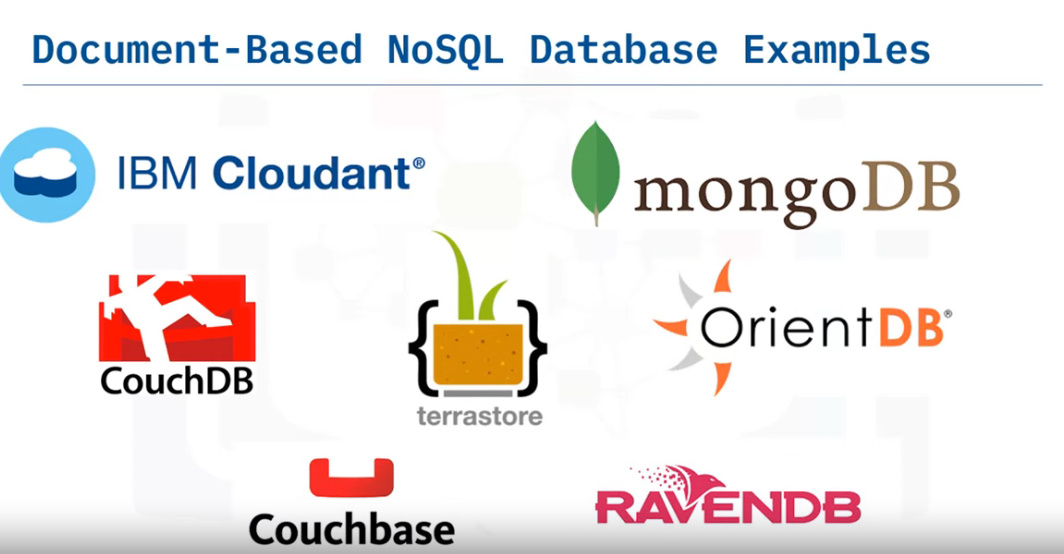
**Use Cases**

****

**Nu sunt bune pentru**

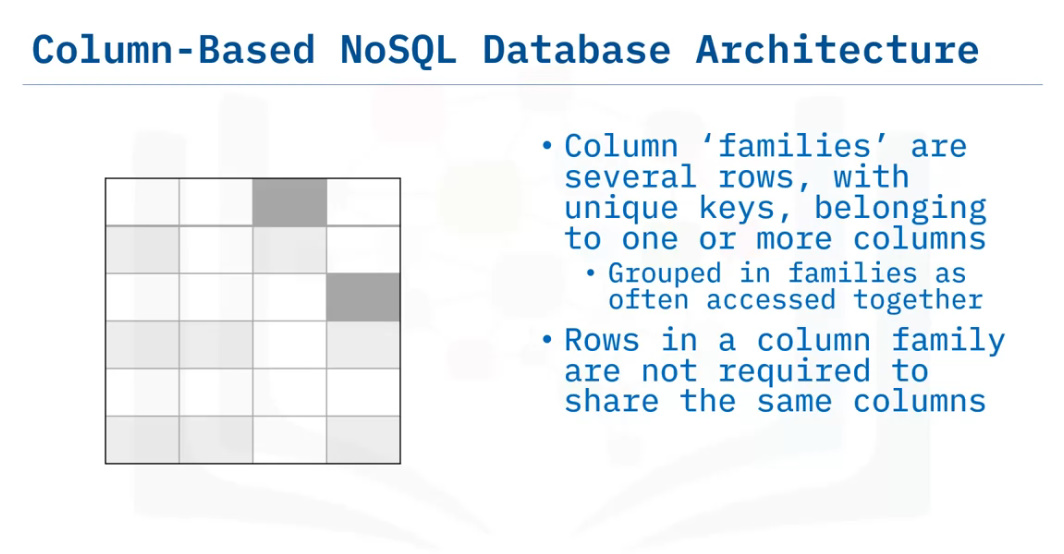


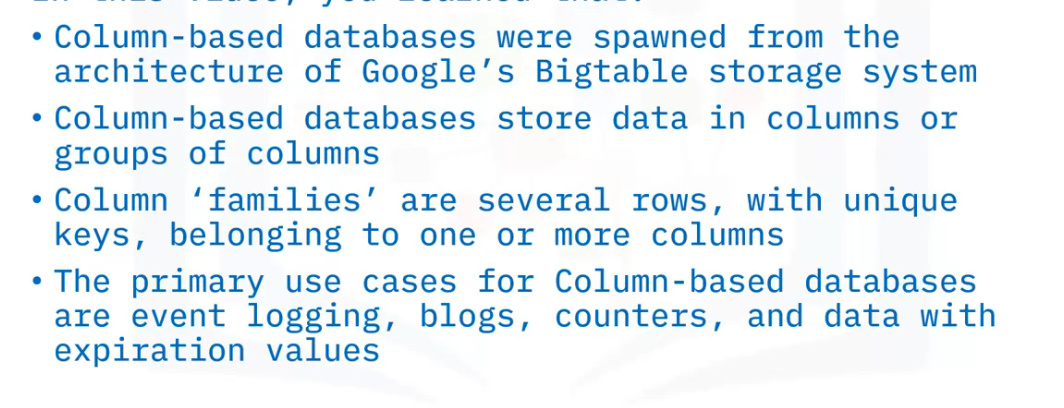
**DMBS**



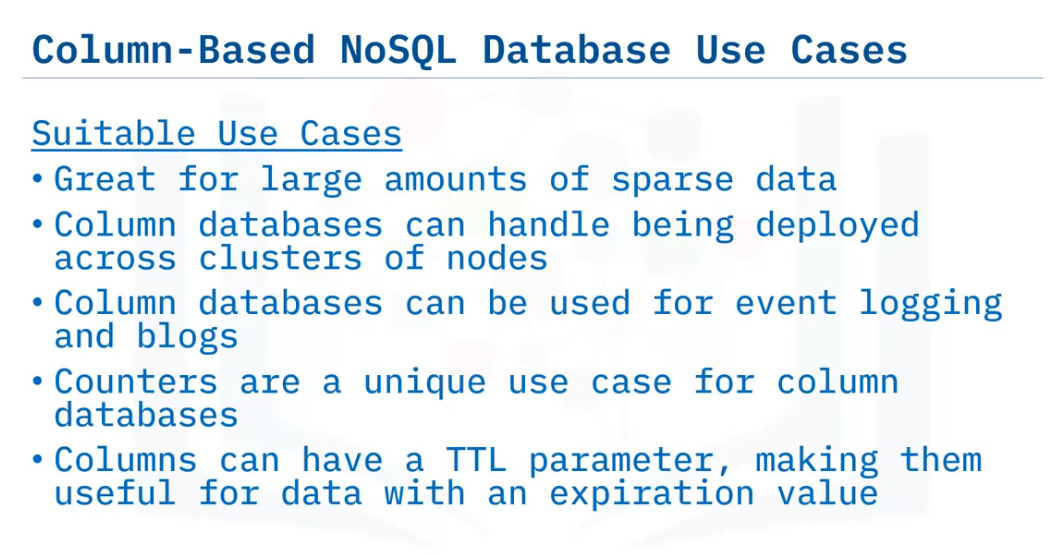
**Column-Based**

* Datele sunt stocate in coloane si grupuri de coloane
* Column families – mai multe randuri cu keys unice care apartin la una sau mai multe coloane
* Liniile intr-o column family nu trebuie neaparat sa aiba aceleasi coloane pe care le impartasesc.

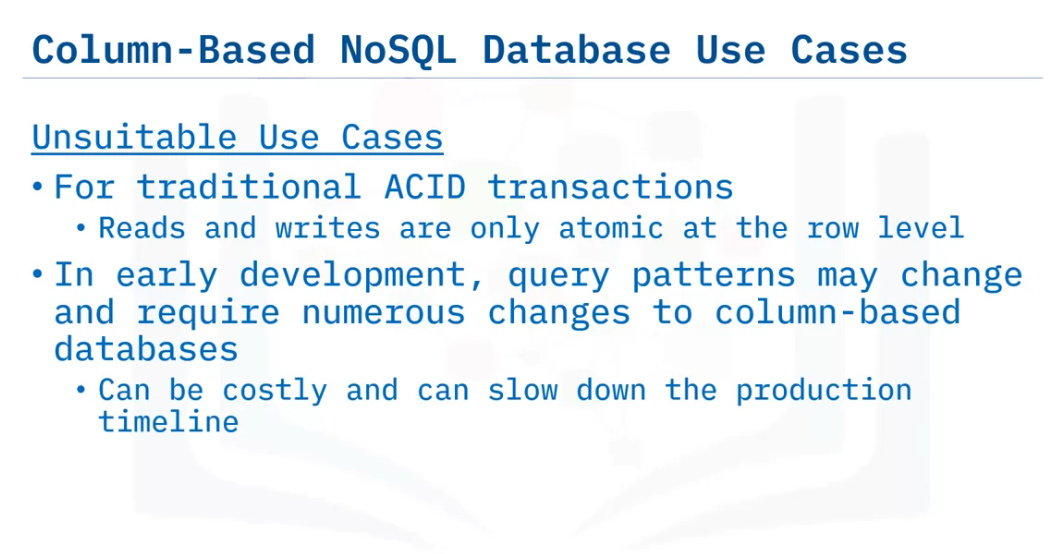




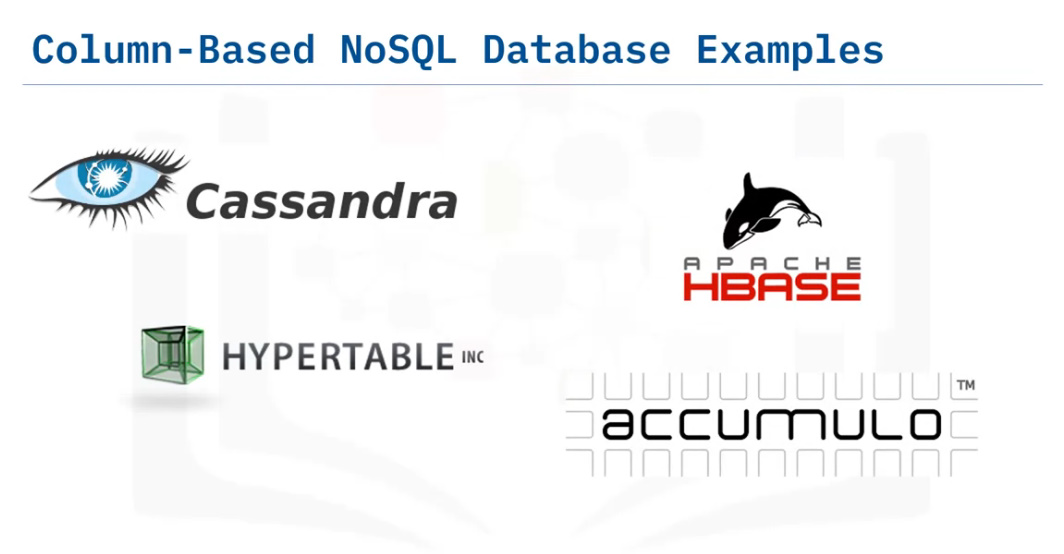
**Cand se foloseste**



**Nu se folosesc**

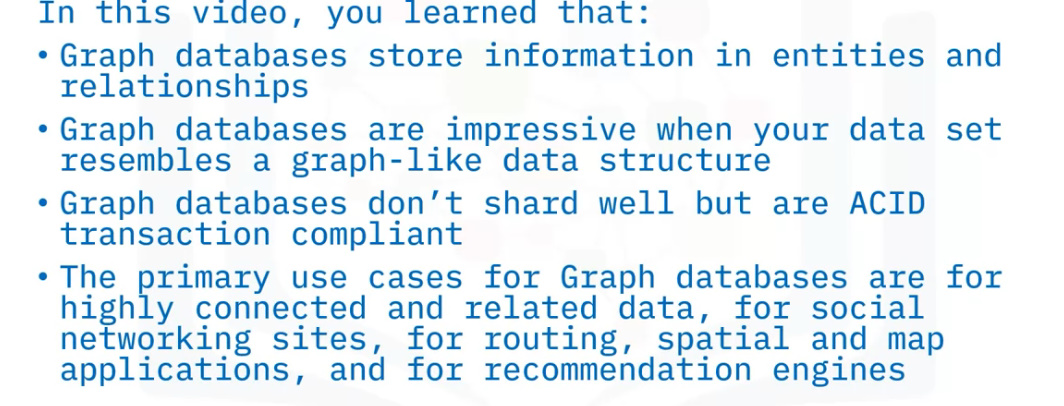


**DBMS**



**Graph**

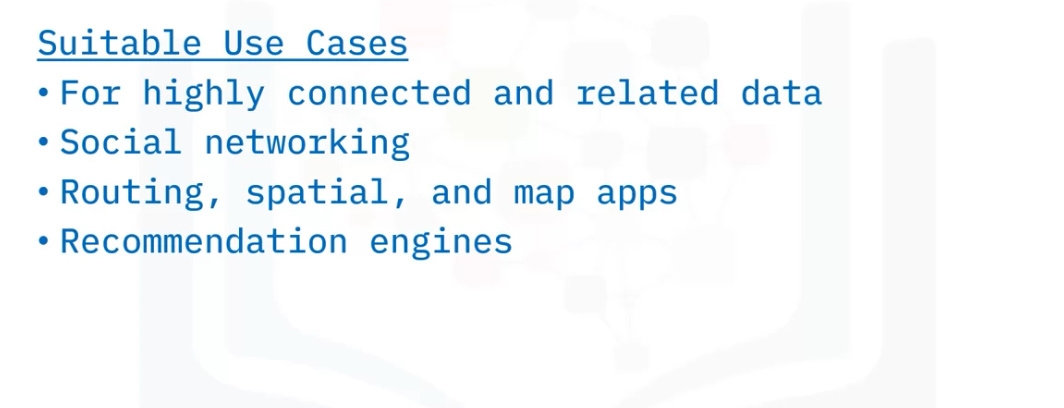
* Este diferita de celelalte 3
* Stocheaza datele in entitati(sau noduri) si relationships(edges)



**Avantaje**

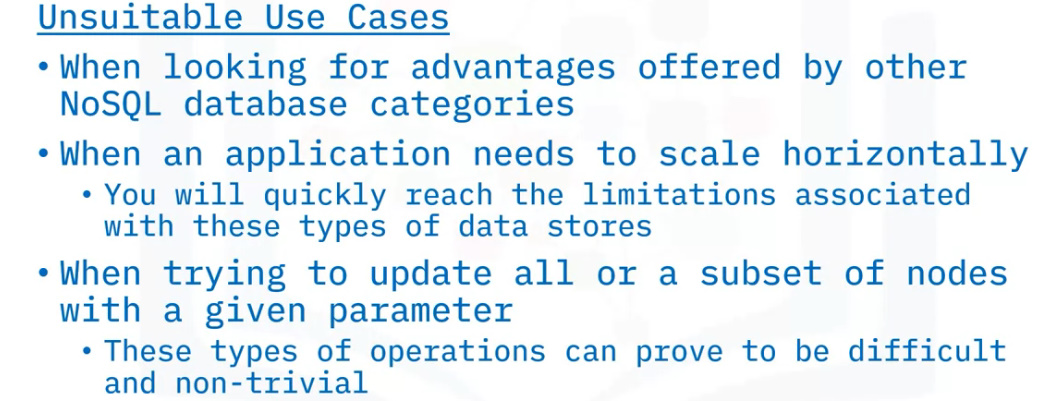
* Sunt bune cand datele sunt stocate in graph data structure
* Sunt bune pentru ACID transactions

**Use Cases**



E buna cand avem prieteni si ei tot au prieteni si tot asa, gen pe facebook

**Nu sunt bune la**



**DBMS**

