

What are database isolation levels (Episode 8)



ALEX XU

MAY 21, 2022



81



4



Share



In this newsletter, we will talk about the following:

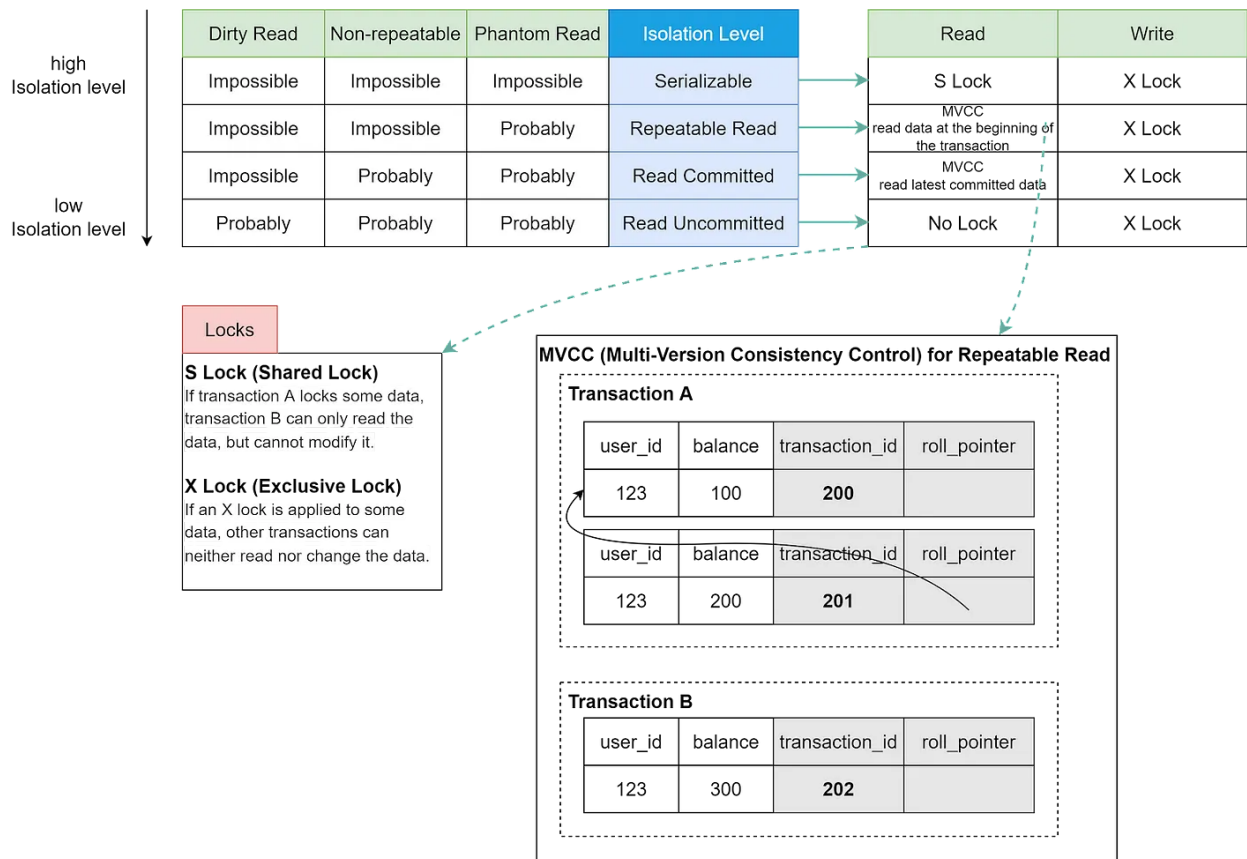
- Database isolation levels
- Log parsing commands
- How does Change data capture (CDC) work?
- ByteByteGo system design Big Archive

What are database isolation levels? What are they used for?

Database isolation allows a transaction to execute as if there are no other concurrently running transactions.

The diagram below illustrates four isolation levels.

Database Isolation Level Illustrated



- ◆ **Serializable:** This is the highest isolation level. Concurrent transactions are guaranteed to be executed in sequence.
- ◆ **Repeatable Read:** Data read during the transaction stays the same as the transaction starts.
- ◆ **Read Committed:** Data modification can only be read after the transaction is committed.
- ◆ **Read Uncommitted:** The data modification can be read by other transactions before a transaction is committed.

The isolation is guaranteed by MVCC (Multi-Version Consistency Control) and locks.

The diagram below takes Repeatable Read as an example to demonstrate how MVCC works:

There are two hidden columns for each row: `transaction_id` and `roll_pointer`. When transaction A starts, a new Read View with `transaction_id=201` is created. Shortly afterward, transaction B starts, and a new Read View with `transaction_id=202` is created.

Now transaction A modifies the balance to 200, a new row of the log is created, and the `roll_pointer` points to the old row. Before transaction A commits, transaction B reads the balance data. Transaction B finds that `transaction_id 201` is not committed, it reads the next committed record(`transaction_id=200`).









Even when transaction A commits, transaction B still reads data based on the Read View created when transaction B starts. So transaction B always reads the data with `balance=100`.

Over to you: have you seen isolation levels used in the wrong way? Did it cause serious outages?

Log parsing cheat sheet

I was doing some log parsing today and totally forgot what commands to use. After some Googling, I found this awesome cheat sheet by Thomas Roccia.

LOG PARSING CHEAT SHEET

 GREP	GREP allows you to search patterns in files. ZGREP for GZIP files. <code>\$grep <pattern> file.log</code>	-n: Number of lines that matches -i: Case insensitive -v: Invert matches -E: Extended regex -C: Count number of matches -l: Find filenames that matches the pattern
 NGREP	NGREP is used for analyzing network packets. <code>\$ngrep -I file.pcap</code>	-d: Specify network interface -i: Case insensitive -X: Print in alternate hexdump -t: Print timestamp -I: Read pcap file
 CUT	The CUT command is used to parse fields from delimited logs. <code>\$cut -d " " -f 2 file.log</code>	-d: Use the field delimiter -f: The field numbers -C: Specifies characters position
 SED	SED (Stream Editor) is used to replace strings in a file. <code>\$sed s/regex/replace/g</code>	s: Search -e: Execute command g: Replace -n: Suppress output d: Delete W: Append to file
 SORT	SORT is used to sort a file. <code>\$sort foo.txt</code>	-o: Output to file -C: Check if ordered -r: Reverse order -u: Sort and remove. -n: Numerical sort -f: Ignore case -k: Sort by column. -h: Human sort
 UNIQ	UNIQ is used to extract uniq occurrences. <code>\$uniq foo.txt</code>	-C: Count the number of duplicates -d: Print duplicates -i: Case insensitive
 DIFF	DIFF is used to display differences in files by comparing line by line. <code>\$diff foo.log bar.log</code>	How to read output? d: Add #: Line numbers c: Change <: File 1 d: Delete >: File 2
 AWK	AWK is a programming language use to manipulate data. <code>\$awk {print \$2} foo.log</code>	Print first column with separator " " <code>\$awk -F: '{print \$1}' /etc/passwd</code> Extract uniq value from two files: <code>awk 'FNR==NR {a[\$0]++; next} !(\$0 in a) {f1.txt f2.txt</code>

 @FROGGER_
THOMAS ROCCIA

Log parsing commands are useful for:

- ◆ Searching patterns in text files
- ◆ Analyzing network packets
- ◆ Parsing fields from delimited logs
- ◆ Replacing strings in a file

- ◆ Sorting a file
- ◆ Displaying differences in files by comparing line by line

Over to you: have you used any command in this list?

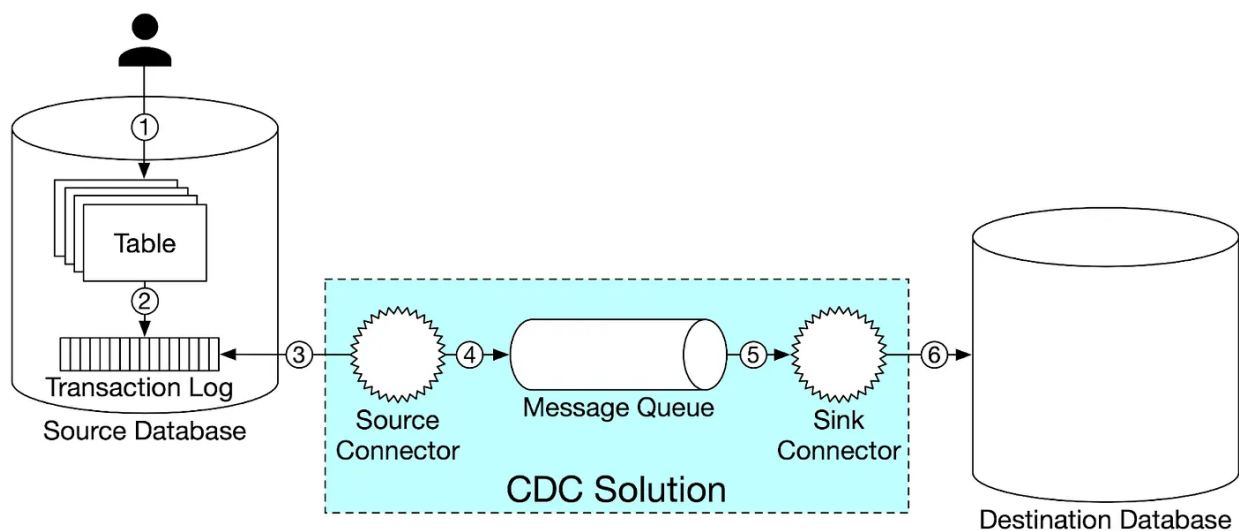
Change data capture (CDC)

Data stored in the database could be interesting to many other data systems, such as analytics, AI, etc. If we have thousands of data systems, do we have to write thousands of converters?

The answer is NO. Change data capture (CDC) is a process that can solve the problem. This is how CDC works:

How does CDC work

Change Data Capture



1. Data is written to the database normally.
2. Database uses the transaction log to record the modifications.
3. CDC software uses the source connector to connect to the database and reads the transaction log.
4. The source connector publishes the log to the message queue.

5. CDC software uses its sink connector to consume the log.
6. The sink connector writes the log content to the destination.

All these operations except step 1 are transparent to the user. Popular CDC solutions, such as Debezium, have connectors for most databases, such as MySQL, PostgreSQL, DB2, Oracle, etc. We only need to set up the CDC link between two databases and the data will automatically flow to the destination.

Over to you: can we use CDC for NoSQL/NewSQL data systems, such as Redis, Cassandra, MongoDB, ElasticSearch, etc?

ByteByteGo System Design Archive

I just put all my technical threads in one big PDF. It has 75 topics and 158 pages!



A little background: I've been consistently posting for 7 months now. With so many people on Twitter reading my posts, I'm extremely grateful.

Here are some sample topics:

- ◆ Why is Redis fast?
- ◆ How to scale a website to support millions of users?
- ◆ How does HTTPs work?
- ◆ What happens when you type a URL into your browser?
- ◆ How to avoid double charge?
- ◆ Why is Kafka fast?

I hope this PDF can be helpful.

Download link:

https://bytebyte-go.s3.amazonaws.com/ByteByteGo_LinkedIn_PDF.pdf

Our Books:

Our bestselling book “System Design Interview - An Insider’s Guide” is available in both paperback and digital format.

Paperback edition: <https://geni.us/XxCd>

Digital edition: <https://bit.ly/3lg41jK>



81 Likes

4 Comments



Write a comment...



Tushar Goel May 23, 2022 Liked by Alex Xu

I think CDC is possible in NoSQL as well.

LIKE (1) REPLY SHARE

...



YoYo Sep 4, 2022

I would like to see more newsletter about database!

LIKE REPLY SHARE

...

2 more comments...

© 2023 ByteByteGo · [Privacy](#) · [Terms](#) · [Collection notice](#)
[Substack](#) is the home for great writing