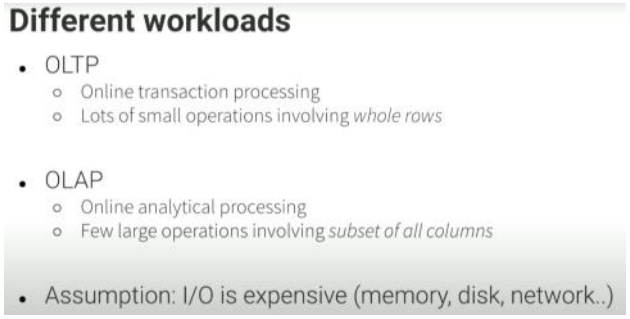
Data Format

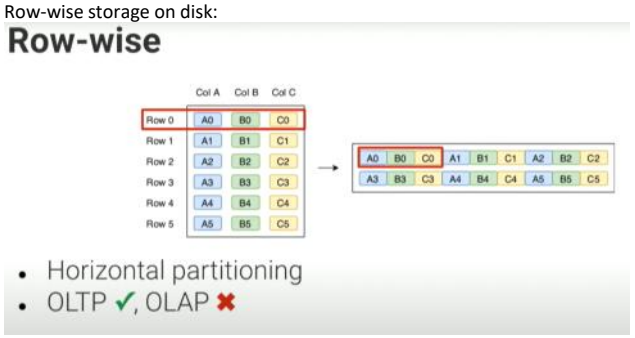
10 November 2022 17:54

Parquet

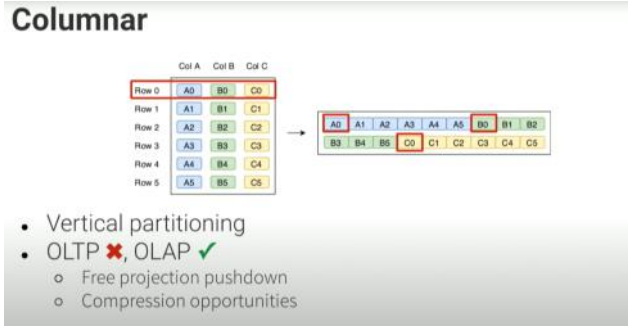
Different Workloads:



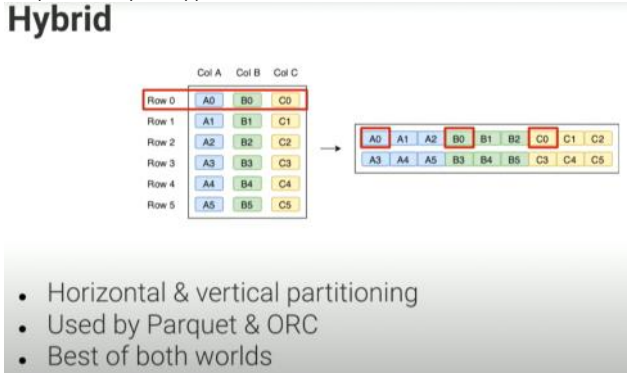
Row-wise storage on disk:



Columnar storage on disk:



Parquet uses Hybrid approach:

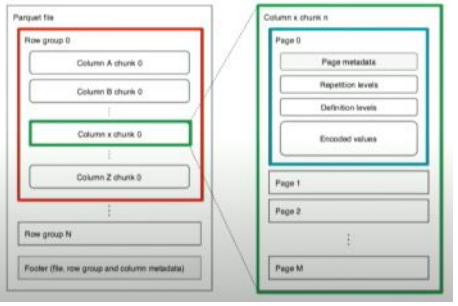


Structure of Parquet:

- **Row groups** - allows vertical partitioning (per row). Each row group has many **Column chunks** (horizontal partitioning, one for each column)
- Pages - where the data is actually stored. It contains **Metadata** (i.e. min, max, count), **Repetition/Definition levels** (for reconstructing nested schemas) , **Encoded values**

Parquet: data organization

- Data organization
  - Row-groups (*default 128MB*)
  - Column chunks
  - Pages (*default 1MB*)
    - Metadata
      - Min
      - Max
      - Count
    - Rep/def levels
    - Encoded values



Encoding in Parquet:

Parquet: encoding schemes

- **PLAIN**
  - Fixed-width: back-to-back
  - Non fixed-width: length prefixed

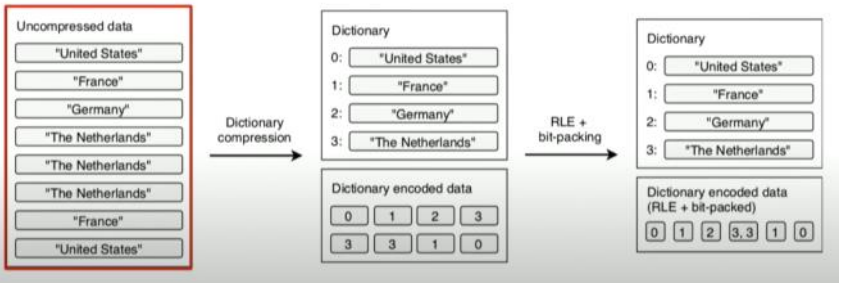
- **RLE_DICTIONARY**
  - Run-length encoding + bit-packing + dictionary compression
  - Assumes duplicate and repeated values

Dictionary Encoding:

- If dictionary becomes too big, it will automatically fall back to PLAIN encoding.

Parquet: encoding schemes

- **RLE_DICTIONARY**



Ways of Parquet optimisation:

- **Dictionary Encoding (using parquet-tools)**
  - Increase max dictionary size (parquet.dictionary.page.size)
  - Decrease row-group size (parquet.block.size)
- **Page Compression**
  - Compression schemes (snappy, gzip, lzo…)
- **Predicate Pushdown**
  - Happens at row-group level

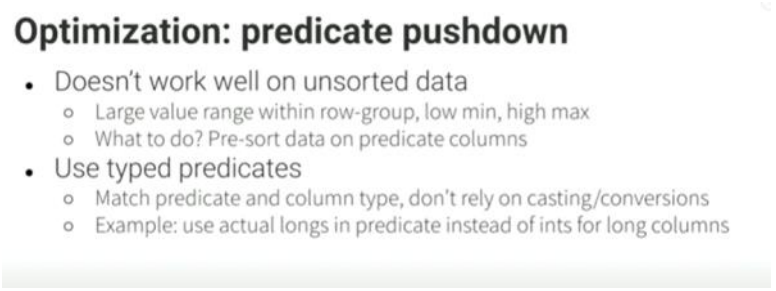
Optimization: predicate pushdown

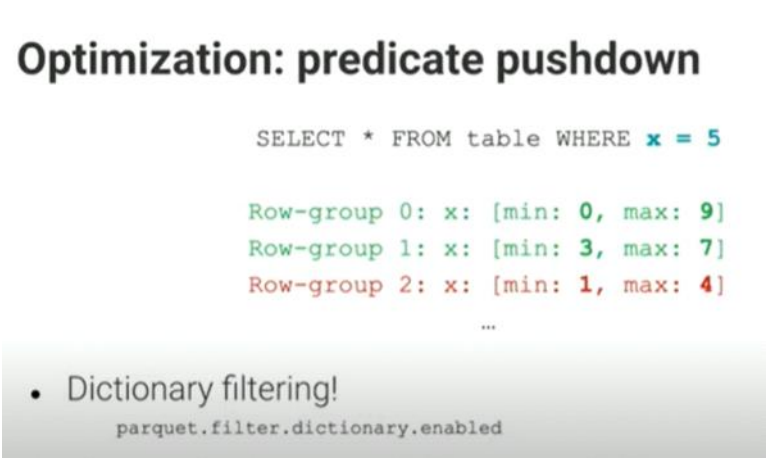
```
SELECT * FROM table WHERE x > 5

Row-group 0: x: [min: 0, max: 9]
Row-group 1: x: [min: 3, max: 7]
Row-group 2: x: [min: 1, max: 4]
...
```

- Leverage min/max statistics

  `spark.sql.parquet.filterPushdown`

Optimization: predicate pushdown





- **Partitioning**
  - Embedded predicates in directory structure (df.write.partitionBy("date").parquet(…)
- **Avoid many small files**
  - For every file (overheads)
    - Set up internal data structures
    - Instantiate reader objects
    - Fetch file
    - Parse Parquet metadata
  - Use Manual compaction
    - df.repartition(numPartitions).write.parquet(…)
    - df.coalesce(numPartitions).write.parquet(…)
  - Watch out for incremental workload output!
- **Avoid few huge files**
  - Footer of row groups become huge. Footer processing not optimised for speed.
  - For example:
    - SELECT count(*) on 250GB dataset, it takes 5 mins for 250 partitions (1GB each) but 1 hour for 1 huge partition (250GB)

What is the difference between parquet files and json files?

How do you test your pipeline?

How do you keep yourself up-to-date with new technologies?

How do you manage credentials?

If you could change anything about what you've made, what would it be?

By no means I have extensive knowledge on the inner workings of ChatGPT, but reading the different voices of ChatGPT, I have a few observations which I believe to be worth

sharing. More

1) As with, the rate of adoption will vary across in different industries.

Industries where repeatable patterns occur (GPT)

2) ChatGPT is a quicker/faster search engine (combines all everything and throws out an output to be the most likely to believed)

3) Is it always true?

I can imagine underneath it is a probablistic model where different answers have chances of being selected.
If most websites (samples) say a certain solution is true, reach a 90% it will probably give that solution 9/10 times. When it is split, 45% and 55%, everytime when you give your answer, it seems like different answers are given and it might not always be the right answers (share video).

In the scientific researches in business context, they want new combinations that have not seen. Search engine input is what is given, but will it give new insights? (not optimising, but some new combintions)

Let's say if I give it a Leetcode question, will it come up with a solution that has not been seen anywhere?

Also, if the input of the underlying models in ChatGPT are websites, does it adjust the weighting based on the true/false nature of the quality of content. If the quality of input data is bad/false, will that ? I think partly because of this reason, questions that involve more objective reasoning, whether an area is a good to live, it will give a neutral output. Most output are factual and have less ambiguity .

With no repercussion, it is a great tool because no immediate actions (monetary terms)
Business, without verifying, cannot risk. Whether to take ChatGPT into last level, or a utility helper for people?
I tend to believe it is a utility helper.

4) Will it jeopardise Google's as a search engine business?

Interesting view from XXX on google having a reputation to maintain. They cannot roll out a tool similar to , ultimately Google is building a platform, an eco-system and Google search engine is only one of them.

It might force Google to rethink whether. I don't know whether Google already has something similar done internally, just not released to the public because they have a reputation to withhold and trust is important. When have you ever doubted whether something is true or not? You can see who writes it, where they are coming from, more extensive. Stackover you see the interactions, people commenting. But ChatGPT filters them out for you, get the most likely output and you miss the context. Then you have to work backwards to verify whether it is true or not?

A machine of truth: is it?