

Quiz 2 Solutions

Q2 Z-order

Suppose you have 4 2D data items: (2,1), (3,2), (4,2), and (1,3). Put them in Z-order. Order items in ascending order, starting with the least significant bit of the first dimension.

Answer: The correct ascending order is (2,1),(1,3),(3,2),(4,2). We work this out as follows (here the “starting” bit is the rightmost – i.e., least significant – bit):

(2, 1) -> (010, 001) = 000110 = 6

(3, 2) -> (011, 010) = 001101 = 13

(4, 2) -> (100, 010) = 011000 = 24

(1, 3) -> (001, 011) = 001011 = 11

Q3 Amdahl's Law

Suppose you have a data science pipeline with 5 operators, which takes 100s to run end-to-end. What is the minimum end-to-end runtime you could ever achieve by further optimizing the operator that currently runs the fastest?

Answer: The fastest operator can take at most 20s to run, which would occur, e.g., if all operators take the same amount of time. By further optimizing it, we could theoretically make it instantaneous, in which case the rest of the pipeline would still take 80s.

Q4 Python Performance

Consider the following python code:

```
#ar1 and ar2 are arrays
#f is an expensive function
out = []
for i in range(0, len(ar1)):
    for j in range(0, len(ar2)):
        if f(ar1[i]) == f(ar2[j]):
            out.append((ar1[i], ar2[j]))
```

List 2 ways you could optimize this code to make it run faster. Consider algorithmic changes, parallelization, etc. Assume the order of the output does not matter.

Answer: One way would be to avoid re-computing f: we can first compute it for each member of ar2 before the double for-loop and cache the resulting values, e.g., in a dictionary, and then within the outer (but outside the inner) loop we can compute it once for ar1[i]. Each comparison within the inner loop can then use these saved values. Another optimization is to perform this work in parallel, by allocating parts of the cross product of the arrays to different workers.

Q5 Partitioning and Joins

Suppose you frequently join three tables, S, T, and R with join predicates $S.a = R.b$ and $T.c = R.d$. These tables are filtered differently in different queries, but for most queries the filtered versions of S are about 1 MB, filtered versions of T are about 100 MB, and filtered versions of R are about 10 GB. Each record in each database is 100 bytes. Assume R is a fact table and S and T are dimension tables (i.e., this is a snowflake schema) and you are doing primary key-foreign key joins.

You are running on a 4 node cluster, with each node having 8 GB of RAM and 1 processor.

Your network can transfer 10 MB / sec and nodes can compare 10 M records/sec.

Which of the following pre-partitioning and join strategies would you recommend, and why?

1. Hash partition S on S.a, T on T.c, and R on R.d. First locally join partitions of T with R via a hash join, then shuffle the result on R.b across the four nodes and join with partitions of S on S.a.
2. Hash partition S on S.a, T on T.c and R on R.b. First locally join partitions of S with R via a hash join, then shuffle the results on R.d across the four nodes and join with partitions of T on T.c.
3. **Hash partition R on R.d and T on T.c. At query time, replicate the filtered version of S across all four nodes. First locally join partitions of R with T via a hash join, then join each partition locally with S.**
4. Hash partition R on R.b and S on S.a. At query time, replicate the filtered version of T across all four nodes. First locally join partitions of R with S via a hash join, then join each partition locally with T.

Answer: S is much smaller than the other two tables, so replicating it over the network at query time will only incur a small penalty (0.1s), compared to the advantage of being able to join R and T locally on each node. Replicating T could also help, but would help less than replicating S because T is larger and therefore it is slower to replicate and slower to join in its entirety with partitions of R.

Q6 Parquet

How large would the Parquet encoding of the following array of 8-byte integers be, in bytes?

20 21 20 19 18 20 21 21 18

Assume that Parquet uses the first element as the reference value, which is stored in parquet using 8 bytes. Subsequent input values are processed in 64-byte blocks. Each compressed block is accompanied by an 8-byte min delta and a 1-byte (max)bits value.

Explain your answer.

1. 72 bytes
2. 41 bytes
3. **20 bytes**
4. 23 bytes

Answer: The first value (20) will be stored by Parquet as the reference value, taking up 8 bytes. The remaining 8 array elements will form a single Parquet block. Let's calculate the array of deltas:

20 ||| 1 -1 -1 -1 2 1 0 -3

The min delta is -3, and will take 8 bytes to store. Let's readjust all the deltas w.r.t -3:

20 ||| (-3) ||| 4 2 2 2 5 4 3 0

We need to represent values from 0-5 inclusive, which requires 3 bits per value. This (max)bits value will take 1 byte to store. There are 8 values, so for the data we need 3 bits * 8 values = 24 bits = 3 bytes.

We have 8 bytes (reference) + 8 bytes (min delta) + 1 byte (maxbits) + 3 bytes (data) = 20 bytes overall.

Q7 Lying with Statistics/Visualizations

Name at least 3 problems with the visualization below and explain briefly how the visualization is misleading:



Answer: The horizontal axis is flipped. The horizontal axis is also not scaled proportionally. The vertical axis does not start at 0. The margin of error is mentioned but not shown on the plot (comparable magnitude to the differences between data points). The thing being plotted is misleading (this is a survey of people's perceptions of violent crime as a problem, not a measure of the incidence of violent crime).

Q8 Multiple Hypothesis Testing

You just became the Head Data Scientist at a new online gaming startup. Your first job is to determine which of your 1000 early users are most likely to buy virtual coins. For each user you know the following attributes:

1. Age
2. Bachelor's degree (yes/no)
3. Master's degree (yes/no)
4. PhD degree (yes/no)
5. Facebook user (yes/no)
6. Instagram user (yes/no)
7. TikTok user (yes/no)

8. Spent over 1h in game (yes/no)
9. iPhone user (yes/no)
10. Purchase amount

Q8.1 P-value I

You have a hunch that age impacts how much your users spend. As a first step, you divide the dataset into young (under 21) and old users (above or equal to 21) and you perform a two-sided t-test with a significance value of $\alpha=0.05$.

If you know that you end up rejecting the null hypothesis after this test, what does the result mean?

1. The chances are greater than 1 in 20 that a difference would be found if the study were repeated.
2. The probability is less than 5% that a purchase difference this large could occur by chance alone.
3. **The probability is less than 5% that a purchase difference this large or larger could occur by chance alone.**
4. With >95% probability older people have an average purchase power of \$13 whereas younger people of \$15.

Answer: Follows from the definition of the p-value.

Q8.2 P-value II

Assume you perform the same two-sided t-test as in the previous question.

If you instead know that the t-test gives you a value of $p=0.021$, do you reject or accept the null hypothesis?

1. **Reject the null hypothesis**
2. Accept the null hypothesis

Answer: The p-value is below our threshold significance value, so we reject the null hypothesis.

Q8.3 Bonferroni Correction

Encouraged by the result, you test 4 additional attributes to see how they impact sales.

Specifically, you also use a two-sided t-test for "Facebook user", "Instagram user", "TikTok user", and "Spent over 1h in game".

The results for all the tests are below (including the one for age):

P-value	Feature Tested
0.005	Spent over 1h in game
0.006	Instagram user
0.015	TikTok User
0.021	Age (young/old)
0.051	Facebook user

However, performing this many tests increases the risk of having at least one type I error. Hence, you decide to do a Bonferroni correction to keep the family-wise error rate at 5%. With the Bonferroni correction, which test differences would be considered significant?

1. Spent over 1h in game
2. Instagram user
3. TikTok User
4. Age (young/old)
5. Facebook user

Answer: With 5 tests, the Bonferroni correction indicates a new significance threshold of $0.05/5 = 0.01$. Only the indicated options have p-values below that threshold.

Q8.4 Correlation Matrix

You decide to do one more thing and create a correlation matrix of all 10 features to quickly look at all potential correlations in the dataset as shown below:

Feature Number	Feature Number									
		2	3	4	5	6	7	8	9	10
	1	C _{1,2}	C _{1,3}	C _{1,4}	C _{1,5}	C _{1,6}	C _{1,7}	C _{1,8}	C _{1,9}	C _{1,10}
	2		C _{2,3}	C _{2,4}	C _{2,5}	C _{2,6}	C _{2,7}	C _{2,8}	C _{2,9}	C _{2,10}
	3			C _{3,4}	C _{3,5}	C _{3,6}	C _{3,7}	C _{3,8}	C _{3,9}	C _{3,10}
	4				C _{4,5}	C _{4,6}	C _{4,7}	C _{4,8}	C _{4,9}	C _{4,10}
	5					C _{5,6}	C _{5,7}	C _{5,8}	C _{5,9}	C _{5,10}
	6						C _{6,7}	C _{6,8}	C _{6,9}	C _{6,10}
	7							C _{7,8}	C _{7,9}	C _{7,10}
	8								C _{8,9}	C _{8,10}
9									C _{9,10}	

Based on the correlation matrix you also discover that a Bachelor degree seems to impact how much time a user spends in the game. You do a one-sided t-test afterwards and the p-value is 0.003. Controlling the family-wise error rate at 5% using Bonferroni, would you consider this as significant enough to reject the null-hypothesis? How would it impact your previous results? Explain your answer.

Answer: In this case, we now have 45 tests to consider. Using the Bonferroni correction, the appropriate significance threshold is $0.05/45 = 0.0011$. The p-value of 0.003 is not below that threshold, so we cannot reject the null hypothesis.

Q9 Compression Techniques

Which compression technique would you choose for each of the following types of data? If you make any assumptions, explain them in the box below each part.

Q9.1

A column in a database of MIT faculty, sorted by name, indicating the year in which they won a Turing Award.

1. Delta encoding
2. Run-length encoding

3. Only store non-null tuples

- 4. Dictionary encoding

Answer: We assume that only a few MIT professors have won Turing awards, so this will be a column of mostly null values.

Q9.2

Quality control sensor data owned by a soda company, stored in chronological order, indicating the exact fill level of each bottle as it leaves the production line.

- 1. Delta encoding**
- 2. Run-length encoding
- 3. Only store non-null tuples
- 4. Dictionary encoding

Answer: The fill level will normally be similar between successive bottles, so delta encoding is appropriate.

Q9.3

A column in MIT Medical's student database, sorted by MIT student ID, indicating the manufacturer of the COVID-19 vaccine received by each student.

- 1. Delta encoding
- 2. Run-length encoding
- 3. Only store non-null tuples
- 4. Dictionary encoding**

Answer: There is only a small number of COVID-19 vaccine manufacturers, so we should encode them in a dictionary instead of storing the full names multiple times.

Q9.4

A timeseries of readings from your IoT door lock, stored in chronological order with 1 reading per millisecond, indicating "0" for "unlocked" or "1" for "locked".

- 1. Delta encoding
- 2. Run-length encoding**
- 3. Only store non-null tuples
- 4. Dictionary encoding

Answer: The door will generally remain in its current state (locked or unlocked) between successive readings, since we sample at millisecond granularity. Run-length encoding is therefore appropriate to encode these long runs of 0s or 1s.