Aditya Kelvianto Sidharta
Aks2266
Introduction to Databases
FInal Exam

Question 1: One way of classifying data is into the categories: structured, semi-structured and unstructured. Give a brief description of each type and an example of each type.

Answer:

Structured data usually contains data that can be stored in a database SQL in a table with rows and columns format. Each of the data in the structured data usually has relational keys between each other, and the data contained in the table can be transformed to fit the pre-designed fields in the relational table. An example of Structured data is Relational Data

Semi-structured data usually contains data that does not fit the schema of a relational database, but it contains organizational properties that can be used to transform them to fit a relational database. Examples of semi-structured data include documents held in JSON Format and XML data

Unstructured data contains data that is not arranged in organizational form. Thus, it would be difficult for us to transform and store them in a relational database format. This data will usually require alternative software in order to process, store and manage the data. Example of unstructured data includes PDF, Video, and Audio files.

References:

- https://www.geeksforgeeks.org/difference-between-structured-semi-structured-and-unstructured-data/
- https://k21academy.com/microsoft-azure/dp-900/structured-data-vs-unstructured-data-vs-semi-structured-data/
- https://www.michael-gramlich.com/what-is-structured-semi-structured-and-unstructured-data/

Question 2: What is the difference between a type, e.g. VARCHAR, INT, and a *domain?* Given an example.

Answer:

Type usually refers to the abstract data type which can be used to specify the type of data that the object can hold: integer data, character data, etc. It is also possible for us to create a Composite Type, where the object that we specify consists of multiple base types defined as default in SQL.

Example: CREATE TYPE comptype AS (f1 int, f2 varchar);

A domain usually consists of a data type and optional constraint that can be used to restrict the set of values allowed.
Example: CREATE DOMAIN postal_code AS TEXT
CHECK(
  VALUE ~ '^\d{5}$'
);

References:

- https://www.postgresql.org/docs/9.3/extend-type-system.html
- https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver15
- https://www.postgresql.org/docs/9.5/sql-createdomain.html
- https://www.postgresql.org/docs/9.5/sql-createtype.html

Question 3: Assume that you know that access to a relational table will be primarily sequential. How would you lay the blocks out on a disk's cylinders, heads and sectors?

If the data is small enough, then blocks should be located in the same track, on sequential sectors. If it's not enough, then it should be located on the same cylinders, as all heads between different platters move together. If it's not enough, then blocks should be located on their subsequent cylinders.

Answer:

Question 4: Most databases do not support hash indexes. What are three benefits of B+ Tree indexes over hash indexes? Express your answer by giving examples of SQL queries better supported by B+ Trees.
(A, B, C) A=2 AND B=7 and C=3.

Answer:

- B+ Tree indexes are able to perform bigger than / less than the comparison whenever they perform selection based on the indexes
- B+ Tree index can be used by the optimizer to speed up ORDER BY operations
- B+ Tree index is able to use any leftmost prefix of the key to finding the rows.

Query:
SELECT A, B, C where A>2 AND B>7 AND C>3
SELECT * FROM *tbl_name* WHERE *key_col* LIKE 'Ad%_ya%';

References:

https://dev.mysql.com/doc/refman/8.0/en/index-btree-hash.html

Question 5: What is the primary benefit of RAID-5 compared to RAID-0 and RAID-1?

RAID-5 setup will allow data reliability, similar to the one implemented on RAID-1. Unlike RAID-0, whenever one of the drives fails, the data within that drive can be inferred from the parity information stored on the other drives without any downtime. Compared to RAID-1, RAID-5 is a more cost-effective setup because it still maintains a bigger percentage of capacity as compared to RAID-1 (Raid-1 only offers 50% capacity of the total drive), and the write speed is faster, as it only needs to calculate the parity instead of rewriting the data twice across different drive

References:

https://www.dataplugs.com/en/raid-level-comparison-raid-0-raid-1-raid-5-raid-6-raid-10/

Answer:

Question 6: Briefly explain the difference between fixed size records and variable size records.

Answer:

Fixed-size records indicate that all records are stored in a same-length-byte, whereas variable size records indicate that each record is stored in a different-length-byte, depending on the size of the record itself. Variable size records are useful whenever you have data such that most of your records are short, with some records that are very long in size. In this example, the fixed record will each uses a length equal to the length of the longest record, which might be wasteful in storage.

Question 7: The database query optimizer may create a hash index to optimize a query. Give an example of a SELECT statement for which the optimizer may create a hash index.

Answer:

```
SELECT *
FROM Student
WHERE StudentName = “Aditya Sidharta”
```

(Assuming that StudentName was used as an index for the Student table)

Question 8: Least-recently-used is a common buffer pool replacement policy. But, sometimes LRU is the worst possible algorithm. Briefly give an example of when LRU is the worst algorithm and why.

Answer:

LRU is the worst algorithm whenever there is an infinite sequence of pages that are accessed, but the number of pages is bigger than the buffer pool available. For example, when the buffer size is 3, and there are 4 pages to be accessed in sequence, A B C D A B C D A B C D …, the page that needs to be loaded will never be in the buffer pool, and thus there won't be any hit at oll

Question 9: Briefly explain the concept of conflict serializable and how it differs from serializable.

Answer:

The concurrent schedule is conflict serializable if the schedule can be transformed into a serial schedule. This is done by swapping operations within the schedule that does not conflict with one another.

Serializability refers to a schedule that always leaves the database in a consistent state. There are two types of serializability - conflict serializability and view serializibility. Schedule that is View Serializable may or may not be conflict serializable. However, all schedules that are conflict serializable must be view serializable.

References:

- https://www.geeksforgeeks.org/difference-between-conflict-and-view-serializability/
- https://beginnersbook.com/2018/12/dbms-serializability/

Question 10: What is a deadlock? How do DBMS transaction managers break deadlock?

Answer:

Deadlock is a situation where two or more transactions are waiting for each other to give up the lock in order to complete their transaction. However, as they are locking each others' tables in order to complete their own transaction, none of the transactions can be completed and the

program is stuck indefinitely. DBMS transaction managers break the deadlock by rolling back all but one of the transactions.

References:

- https://www.geeksforgeeks.org/deadlock-in-dbms/