University of California Irvine
Donald Bren School of Information and Computer Sciences
Department of Computer Science

CS 222 - Principles of Data Management

Project Report - Part 02

Sky Faber **Ekin Oguz** **Cesar Ghali**

1 Catalog

The database catalog contains two self-describing tables, tables and columns. Tables table contains information (table name, file location, file type, version) about each table created in the database. The version indicates the latest version of the table in the database. The columns table contains information about each column in all the tables in the database. The information stored is column name, table name, position, type, length, nullable, version. The version contains what version the table is in. Whenever an attribute is added or dropped from the table, the version field is increased. In addition, the version of each record is stored along the record in the database. The idea behind versioning is that when a table is altered, all old records are untouched. When a tuple is read or the table is scanned, all old versions will be reformatted in the most recent version of the table.

2 Record Format

Each record with n fields is formatted as follows:

- The first byte is a boolean specifying whether this record is a forward pointer or not.
- The second byte is the version of this record (more about this later).
- $n + 1$ offset pointers (each one is two bytes). Each one points to the beginning of the corresponding field and the last offset pointer points to the end of the record.
- After the fields directory, all n fields are concatenated.

Using this format accessing any filed takes $O(1)$ operation. It is also not necessary to store the length of the field. Instead, it can be calculate on the fly with $P(1)$ operations. The offset of the first field is read, then the end offset of the field is read (value of first offset - 2) which indicates the length of the field.

3 Page Format

Pages in the database contain records of variable length. All records are packed to the begin of the page. The last two bytes of the page contains the offset of the beginning of the empty space on page. Right before these two bytes, the record directory is located. This directory contains in its first two bytes the number of records in the directory. The rest of the directory contains two bytes offset for the beginning of the records on the page.

The first page contains information about the free spaces on each page of the file. The first two bytes of the page contains the number of pages on the file. Then each two bytes contains the amount of free space on each page.

3.1 Operations on records

1. Record Deletion: when a record is deleted, its slot in the records directory is assigned the value `0xFFFF` rending this slot to be empty. In addition, when a record is deleted, its physical location on the page is kept empty and the free space on the first page is updated.

2. Record Addition: when a record is added, the directory is scanned for the first empty slot (marked with `0xFFFF`) to be used. If such a slot could not be found, a new slot is added to the end of directory and the number if records is increased. Note that the offset where the record is stored on the page is obtained from the last two bytes of the page (the empty space offset). Also the free space on the first page is also updated to reflect the addition of a new record to this page.
   The addition algorithm consults the first page of the file looking for a page with free space. When such a page is found and the record cannot fit at the end of the page (starting where the free space pointer points) the page is reorganized but without changing the slot numbers. If no pages have free space, a new page is added to the file.

3. Record update: updating a record is basically deleting that record and add it again but in the same page and at the same slot number. If there record cannot fit on the page, a forward pointer is created. The forward pointer contains the alien page number and the slot number of this record. Note that when a record is updated, it should be located on the same page and at the same slot number as before, so a forward pointer is updated to point to the place where the record is stored in case that page was full. This whole operation is transparent for any upper layer.

4 Altering the database

The columns table of the catalog contains information about all the columns in each table in the database. When a new column is added to a table, the latest (in terms of version) catalog information about this table is read, then reinserted with a new version number. When a column is deleted, all the old versioned information is reinserted with a new version but without the deleted column. Note that reading records or scanning a table will take care of converting all the filed to the latest version on the fly. Records are not reformatted and stored in the file unless the whole table is reorganized.

5 Table Reorganization

When a table version reaches a certain predefined threshold, e.g. 10, the whole table is reorganized. At this point, all records are read, converted to the latest version and stored in the file without changing their RIDs. If a record cannot fit on the page that it was located in, a forward pointer is used. Unfortunately, this feature is not implemented.