**MAKERERE UNIVERSITY**

**COLLEGE OF ENGINEERING, DESIGN, ART AND TECHNOLOGY**

**SCHOOL OF ENGINEERING**

**DEPARTMENT OF COMPUTER AND ELECTRICAL ENGINEERING**

**CMP2101: SOFTWARE ENGINEERING**

**NAME: MUHUMUZA HILLARY**

**REG No: 12 / U / 644**

**STUDENT No.: 212000366**

**SIGNATURE: ..........................**

**QUESTION**: COMPARE AND CONTRAST THE PROS ACHIEVABLE THROUGH VERTICAL STRUCTURAL PARTITIONING AND HORIZONTAL STRUCTURAL PARTITIONING.

STRUCTURAL PARTITIONING.

**Horizontal partitioning:** horizontal partitioning describes separate branches of the modular hierarchy of reach major program function. The control modules (represented by the shaded boxes in Fig.1) are used to co-ordinate communication between and execution of the functions. The easiest approach to horizontal partitioning describes 3 partitions - input, data transformation often called processing and output. Partitioning this way provides the following **benefits:**

  i.      Results in software that is easier to test and maintain.

  ii.      Results in fewer propagation side-effects.

ii.      Results in software that is easier to extend.

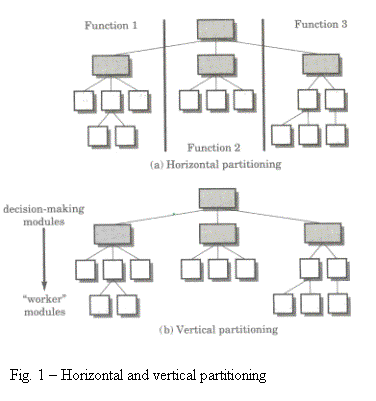
This is because the major functions are ‘decoupled’ from each other. One disadvantage is that more data has to be passed across module interfaces. This can cause the server to complicate the overall control flow of the program especially if processing requires rapid movement from one function to another.

**When to use horizontal partitioning**?

* Certain rows are accessed more often than the rest of them.
* The entire row is useful.

To give an example, let’s consider customer purchase details. You’ll need to access the current year customer information more frequently than previous years.

So, let’s say you maintain purchase details for 5 years. You’d be better off splitting the information year wise.

.

**Vertical partitioning:** in vertical partitioning,the control (decision making) modules are located at the top, and work is distributed top-down in the program structure. That is, top level functions perform control functions and do little actual processing, while modules that are low in the structure perform all input, computation and output tasks. As changes to programs usually revolves around one of these three tasks there is less likelihood that changes made to the lower modules will propagate (upwards) making this partitioning strategy more maintainable. Note however, that changes to control modules will have a higher probability of propagating side-effects.

**Advantages:**   
- good at dealing with changes.  
- easy to maintain the changes.  
- reduce the change impact and propagation.

**When to use vertical partitioning?**

* You don’t need the entire row. That is, a few columns are more frequently accessed than the rest of the columns.
* All rows are equally likely to be accessed.

So, if we had a table with 10 columns and only 4 of them were frequently used, we can split the table into 2 tables, one with;

* 4 columns and a PRIMARY KEY field
* 6 columns and a FOREIGN KEY field

We have this notion of [PRIMARY KEY](http://en.wikipedia.org/wiki/Primary_key) and [FOREIGN KEY](http://en.wikipedia.org/wiki/Foreign_key) to map a row in one table, uniquely to a row in the other table.

* <http://msdn.microsoft.com/en-us/library/ms178148.aspx>
* <http://dev.mysql.com/tech-resources/articles/performance-partitioning.html>
* <http://www.youtube.com/watch?v=HAzeJTgFwDc>
* <http://www.zdnet.com/videos/whiteboard/database-partitioning/153905>
* strthoughts.wordpress.com
* www.cavehill.uwi.edu
* productdevelop.blogspot.com