??

???????? IBM DB2 ??

???????? http://db2commerce.com/2015/07/01/db...linux-systems/

DB2 doesn't have any defaults for filesystems because that is an OS-level thing. However, there are a few sanity checks and some recommeded separation when installing DB2 on Linux and UNIX operating systems. These are my best practices for installations, in general terms.

Software Location

The default location for installing the DB2 code in /opt/ibm/db2/V10.5 (replacing V10.5 with whatever version). Yes, I typed that from memory without having to look it up. The default location works fine, though I would recommend that /opt is mounted as its own filesystem and not just part of root. There are reasons to put the code in other places, but I wouldn't unless you have a technical reason to do so. If you're using db2_install (my favorite installation method), it will prompt you for the install location.

Don't Create Instances in / or in /home

After software installation, the next location that DB2 uses is the home directory of the instance owner. Before creating the DB2 instance (either using db2setup or using db2icrt), you need to verify the primary group and the home directory of the DB2 instance owner. DO NOT place the home directory of the db2 instance owner on / or /home. The reasoning for this is because those locations may quite easily be filled up by other users on the server, and if the instance home fills up, DB2 becomes unusable. We don't want a random user transferring a giant file and crashing the database. The home directory of the DB2 instance owner cannot easily be changed after instance creation.

Other Recommended Filesystems

For a minimal usage system, I can live with only two filesystems as described above – /opt and some filesystem for the home directory of the DB2 instance owner. But if I'm setting up a production system to the best reasonable design, I also include separate filesystems for at least the following.

Two Data Filesystems

I place the database home directory in a data filesystem. I like to have two just because it makes sense to me to start with two, even if they are both part of the same storage group.

The easiest way to configure a new database to use these filesystems is on the **create database command** – specify **ON /path1, /path2 DBPATH ON /path1**. I will also frequently set the **DFTDBPATH DBM CFG parameter** to the first data path.

Active Transaction Logs

Active transaction logs are critical to database performance and also to database functioning. Active logs should be on the fastest storage you have, and you should train operations and other people who might touch your database server to NEVER touch the active transaction log files. I have seen a production database crashed by a more junior operations person compressing an active transaction log.

To direct active log files to a new path, you'll have to set **the database configuration parameter NEWLOGPATH** and deactivate/activate your database for the change to take effect

Archive Transaction Logs

If you have TSM or another location for Archiving transaction logs that is best and safest. If you are archiving to disk, these MUST be in a different directory from active transaction logs, and preferably in a separate filesystem. The logic for a separate filesystem is that this gives you an extra layer of protection – if you're monitoring for full filesystems, you will catch a logging issue when it fills up your archive log filesystem, and hopefully have time to address it before it also fills your active log filesystem and makes your database unavailable.

To archive transaction log files to a path, you have to set **the LOGARCHMETH1 db cfg parameter to: DISK:/path**. If you're setting LOGARCHMETH1 for the first time, you may also be changing from circular logging to archive logging, which requires an offline database backup, so be cautious.

db2diag/db2dump

The db2 diagnostic log and other diagnostic files by default will be in $INSTHOME/sqllib/db2dump. I like to have them in another filesystem – this ensures that no matter what else fills up or what other filesystem level problems are encountered, I should still get the error messages from those issues.

The location for this data is changed using **the DIAGPATH parameter in the DBM cfg**. It can be updated with an UPDATE DBM CFG command, and changes to it take effect immediately.

Backups

I like to take backups to a different filesystem. If your filesystems are fully separate I/O paths, this can have some performance benefits. But the real reason is because the backup location is the location you're most likely to fill up, and you don't want a filesystem filling up because of a backup and causing an availability issue for your database.

Specify the backup filesystem on any **BACKUP DATABASE** commands you issue, including those from scripts.

Scripts

I have seen a script go a bit haywire and capture more data that it was thought it would, and fill up the filesystem it is writing data to. For this reason, I like to keep my administrative scripts and their output on a separate filesystem from everything. This makes it much harder for a scripting mistake to cause a database availability issue.

Others

Obviously there are other filesystems that particular designs may call for – such as a shared filesystem between two HADR servers when doing loads with copy yes. But the above are true to just about every DB2 installation.

Separating I/O

This post does not cover storage design in depth. Usually the storage I get is a black box. A client allocates the space locally on the server or on a SAN, and I have a very hard time finding out the storage details. If I actually get to specify anything about where the storage goes or what is separate, my first priority is to separate out my active transaction logs and put them on the fastest storage I have, separate from everything else. If I have more specificity available, I like to separate my two data filesystems from my backup filesystem. After that it's all icing. In only a very few cases do I see indexes and data separated any

more. When I started as a DBA 14 years ago, that was a core tenant of DB database storage design.

It is Only Temporary

I cannot tell you how many stupid things I have been asked to do in the name of "It's only temporary!" or "It's only a development environment", only to have those things become permanent fixtures in a production environment. Just today, I installed DB2 on root, with no separate filesystems at all, after making it perfectly clear I thought it was a horrible idea. My exact words were:

> If you want me to move forward with the understanding that a significant percentage of our clients come to us just because of this kind of mis-configuration and the stability issues it causes, then I can. But I want you to understand what a bad idea I think it is.

There really is no such thing as "It's only temporary" – design correctly from the start and the problems that you encounter are much easier to deal with.