

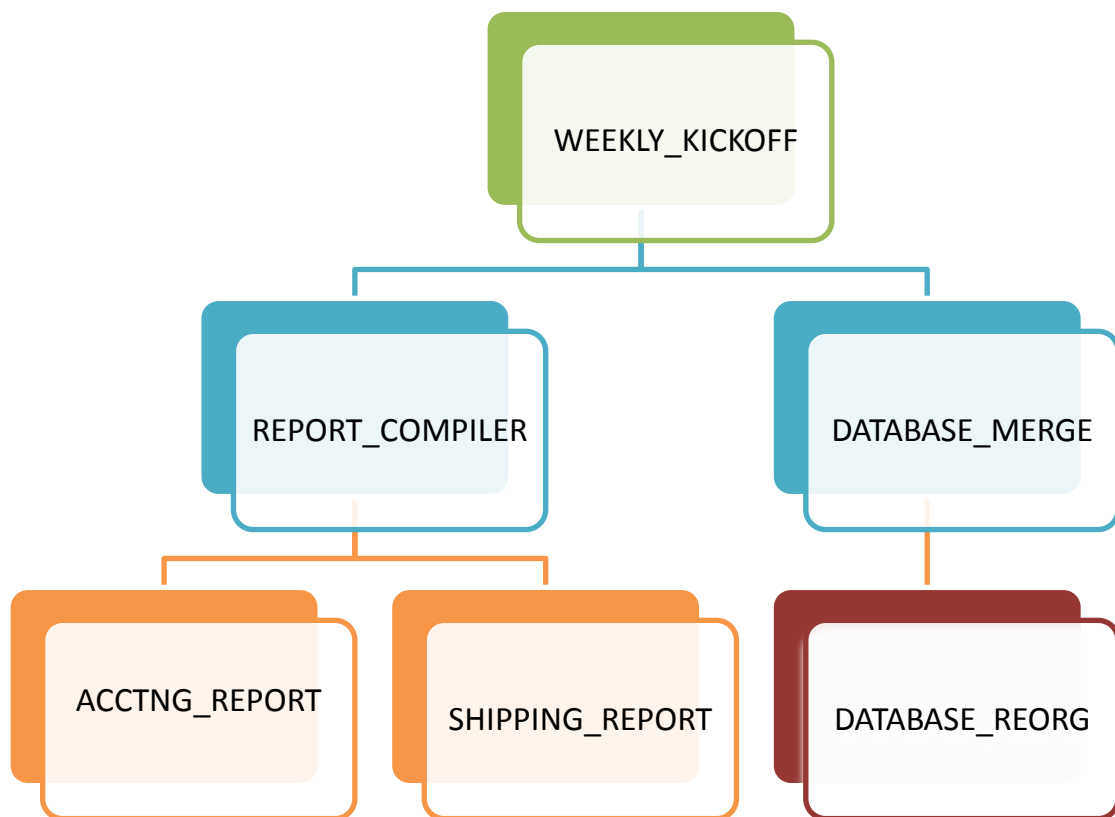
Understanding Control-M Prerequisite Conditions

Robert Stinnett, July 2010 – www.robertstinnett.com

Prerequisite conditions in Control-M help link together jobs such that jobs cannot execute before another job has completed. This prerequisite job may have executed on the same day, or weeks ago, depending on how the job flow is setup. Conditions help you gain control over your job and workload, and can also be a powerful tool to help you decide when to run – or when not to run – based on other events that have taken place in the Control-M environment.

So Just what is A Prerequisite Condition?

Let's take a look at the following example for a more visual explanation.



Here we have a typical job flow. I've color coded it to make it easier to follow along. This job flow is a weekly flow, but takes several days to complete. It's ordered up on Monday and some of the jobs farther down don't complete until Thursday of the same week.

The green job WEEKLY_KICKOFF has no prerequisite conditions. In other words, it is not dependent on anything else finishing first before it can run. Keep in mind the job must still meet its quantitative and control resource requirements, as well as meet its time window and have its node available for execution. However, assuming all those are met it can run without any dependency on a job before it completing.

The blue jobs, REPORT_COMPILER and DATABASE_MERGE both depend on WEEKLY_KICKOFF completing before they can run. In this instance, two jobs are dependent on a single job before it completing and posting its OUT CONDITION. Neither job will begin executing until WEEKLY_KICKOFF_OK with the order date matching the same order date as REPORT_COMPILER and DATABASE_MERGE exist in the prerequisite conditions table. We'll talk more about order date and dates in general in a bit.

The same principle holds true for ACCTNG_REPORT and SHIPPING_REPORT. They will not be able to execute until REPORT_COMPILER is finished; DATABASE_REORG cannot begin executing until DATABASE_MERGE is completed.

Keep in mind we said earlier that this job flow can take several days to complete because some of the jobs run for a long period of time. This is where dates come into play.

I Got A Hot Date!

If you could peek inside the database that Control-M uses to store prerequisite conditions you would see something like this for our conditions:

```
WEEKLY_KICKOFF_OK 0520
WEEKLY_KICKOFF_OK 0513
REPORT_COMPILER_OK 0520
REPORT_COMPILER_OK 0513
DATABASE_MERGE_DONE 0520
DATABASE_MERGE_DONE 0513
DATABASE_MERGE_DONE 0506
```

Confused about where all the conditions with the same name came from? It's all about the date the original job was ordered up. Control-M stores conditions with the order date added as a field so that it can distinguish between different job flows and different dates. That way if two job flows that are exactly the same, but from different days of the week, are running it knows which conditions belong to which based off the date field.

There is also a special condition type called STAT (short for STATIC) that you can use. However, its use should be limited to situations where you have an absolute need for a static condition to exist (and then you may want to give consideration to using a quantitative resource instead with a value of 1 or 0 to control the flow).

Not Today, Maybe Tomorrow or Was It Yesterday?

Some job flows have to be able to look forward and backward in time, or they may just need any condition for any date posted. Luckily, Control-M has you covered.

When you specify what date you want the condition checked against in your CONDITIONS tab for your job definition you are give several options:

ODAT – The original order date for the job flow (most commonly used).

PREV – The previous order day (good for when you don't want a job to run until you are sure the previous day/week/etc. has finished up)

NEXT – The next order date. Most commonly used with OUT conditions to set a condition with the next order date in it.

\$\$\$\$ or **** - Wildcards. Match any date!

A Condition by Any Other Name

So do conditions have a standard naming format? They don't, but that doesn't mean you shouldn't follow one! Conditions can be any name that you wish. You will find that most people stick to the format of "JOBNAME-ENDED" or "JOBNAME-OK". However, there may be situations where you want to post a condition with a different name than the job name.

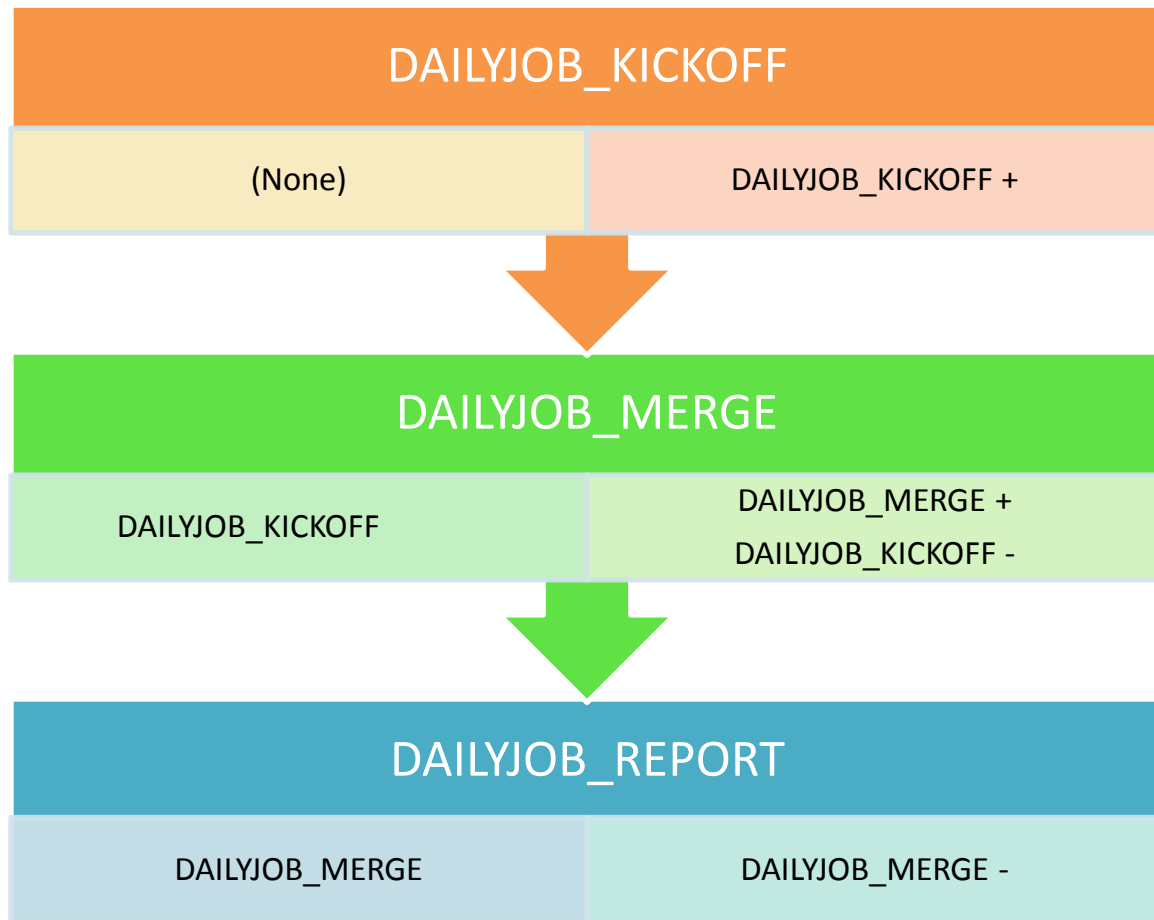
Some of the reasons for doing this include posting a flag condition that gets used throughout many jobs, or a condition that is used to control whether or jobs are allowed to run or not based off some other event.

For example, I use conditions with different names to act as flags for weekly and monthly flows. I look for the presence of that flag condition to tell me if it is OK or not to process other jobs that depend on the weekly/month flows running successfully.

Mr. Clean Cleans Anything – Including Conditions!

So now that you've posted your conditions out, and used it in another job what do you do it? Good practice says that you clean it up to keep old conditions from hanging around the system and to make reruns easier. For example, let's take a look at this simple diagram of a 3-step job flow that runs each day.

The left hand column indicates prerequisite conditions that must be satisfied for the job to run; the right hand columns shows conditions that are posted after job completion.

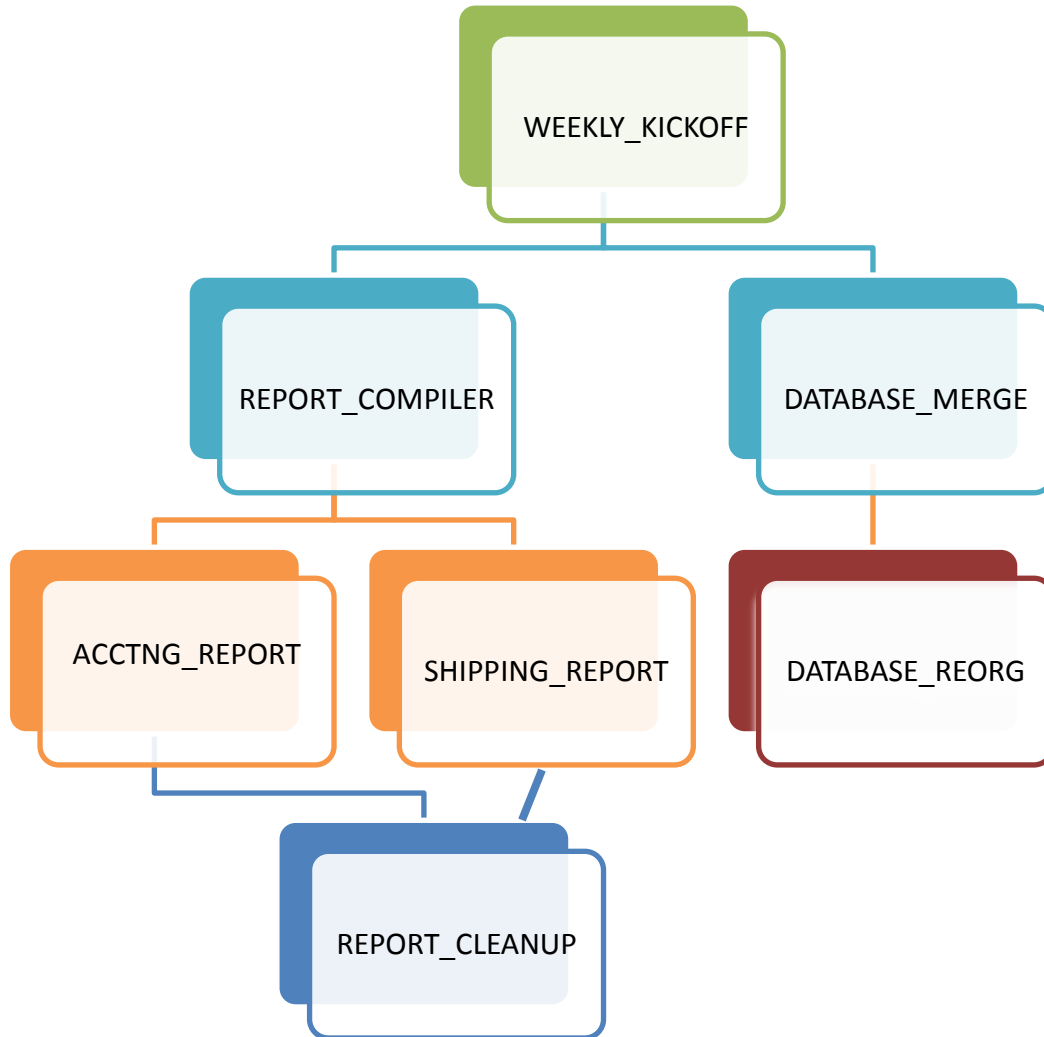


As you can see, we are cleaning up all of our in conditions when the job finishes. Anything that was needed coming in is cleaned up after the job completes.

This works for job flows for the one above where there is a “straight line” relationship. However, if you have job flows that branch around and in conditions are used for multiple jobs you want to be careful not to clean up conditions that are still being used.

If you find yourself in a situation where you have multiple jobs using the same condition one way to clean up this common condition is with the use of a dummy job.

Let’s take a look at an example of this.



This should look familiar – it’s the same job flow we started out with at the beginning of this document. We have the job REPORT_COMPILER that posts a condition that is used for the prerequisite conditions for two jobs. We add a third job after those two to handle the cleanup of that condition.

The REPORT_CLEANUP job is a DUMMY job with the following OUT conditions:

REPORT_COMPILER -
 ACCTNG_REPORT -
 SHIPPING_REPORT -

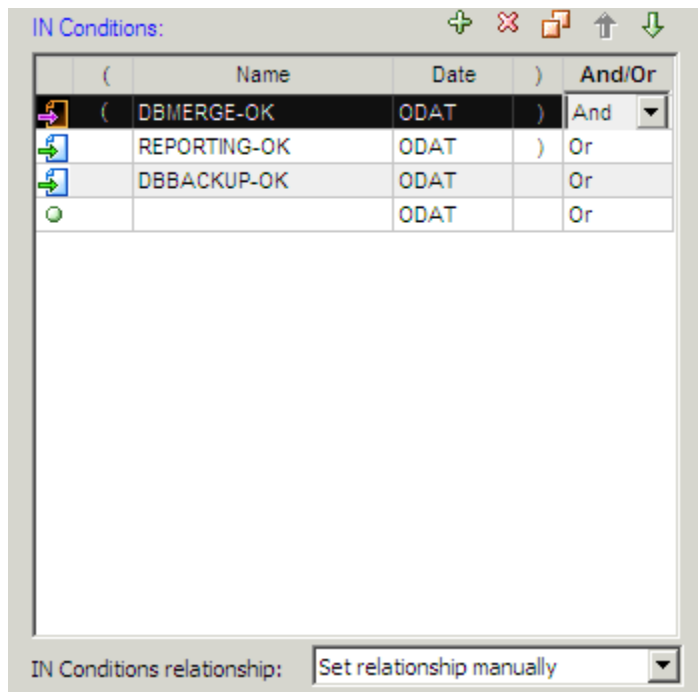
In order for that job to run both ACCTNG_REPORT and SHIPPING_REPORT must have completed. This dummy job simply cleans up the conditions for the REPORT_COMPILER job, plus the ACCTNG and SHIPPING report jobs.

One, Both, Many or None at All?

Just when you thought it was safe to start writing jobs here we go changing it all up again. When dealing with in conditions you have the option of telling Control-M if the conditions you list should be ANDed together (all must be true), ORed together (one must be true) or you can either set the relationships manually (I need this one AND that one to run, or I just need this one to have completed).

This is all controlled through your job definition form on the Conditions tab. AND and OR are pretty self explanatory. By default Control-M uses the AND relationship.

The fun starts when you get into the manual setting of relationships. Let's take a look!



Here I've told Control-M that I need the conditions DBMERGE-OK AND REPORTING-OK in other to run. These both need to be present (so I put them in parenthesis) before this job can run OR the DBBACKUP-OK condition must be present.

So I've created two ways for this job to kick off, either I need two conditions at the same time, or I just need one condition from a different job.

Setting the relationship manually gives you a more granular control over your job flows, but remember you also have to maintain these jobs as the months and years roll by. Complex condition sets can make ongoing maintenance of jobs a nightmare.

Questions, Comments, Funny Jokes?

Hopefully this document has helped explain just exactly what conditions are in Control-M and helped you learn how you can use them to control your job flows. Through the use of conditions and basic AND/OR logic you can create sophisticated job flows that help your systems run at their peak instead of wasting precious hours and minutes by trying to simply “run it all one after another”.

If you have any questions, comments or suggestions on how to improve this document – or maybe you have another topic you’d like for me to cover – please don’t hesitate to contact me.

Robert Stinnett

www.robertstinnett.com

Robert@robertstinnett.com