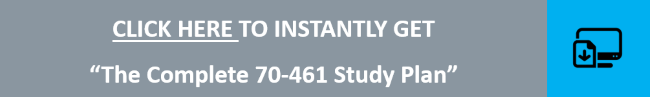
**Ten Things a Junior DBA Should Learn**



There is more to being a Junior DBA than knowing SQL.  The DBA works at the intersection of the database, server, operations group, and developers.  A DBA must understand concepts from all these facets of IT as well as be able to draw upon knowledge of their production environment to troubleshoot performance, hardware, and software issues. Below is a list of the ten topics I feel every entry level DBA should understand.  The list I created stems from my experiences working with databases as both a DBA, Developer, and Manager.   When looking to hire entry level DBA’s, the interview questions I ask are drawn from these areas. If you’re looking to start a job as DBA, then you’ll want to be familiar with these topics.

**Backup and Restore**

Any DBA worth their salt should know the DBMS’ (Database Management System’s) built-in methods to backup and restore data, such as using Oracle Recovery Manager, but in addition to these built-in utilities, it also makes sense to understand what third party offerings exist. Enterprise backup solutions are used in many larger IT shops.  Be familiar with products such as [NetBackup](http://www.symantec.com/business/support/index?page=content&id=DOC6479) or [NetApp SnapManager](http://www.netapp.com/us/products/management-software/snapmanager-sql.aspx). As a junior DBA it would be impressive if you knew these tools existed and that not all backups are created equal.  That is to say, just because you back up the database files, doesn’t mean get a good backup… in fact, you didn’t.

[](http://www.essentialsql.com/studyplanarticle)

**Basic optimizations**

It is important to know when to recommend when an index should be created.  You should know some basic indexing strategies.  When are clustered indexes appropriate?  When should you use a covered index? Also know how your database optimizer works.  Does it rely on special table statistics?  How do you update those? Know what it means to reorganize tables and indexes.  When should they be reorganized, and what can you do to automate the process?

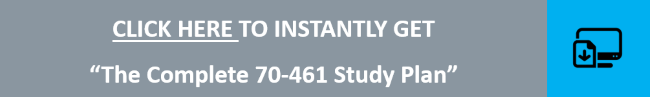
**[](https://277dfx2bm2883ohl6u2g3l59-wpengine.netdna-ssl.com/wp-content/uploads/2014/06/Software-Developers.jpg)**

**Developers**

Software developers can make or break a database.  It is important that you can work with others to help them write efficient queries.  You need to help them understand that one call to a database is much more efficient than one thousand!  In most cases it is faster run one query that returns a 1,000 rows, than it is to call 1,000 queries that return one row each. As a DBA you should help them understand when it is better to perform processing on the DBMS rather than in code.  Pulling volumes of data across your network to perform a sum is most likely slower than writing a query with an aggregate function.

**Storage Systems**

In most databases the main bottle neck is disk access.  Understanding where your database is stored and how the DBMS accesses the physical data is important.  Is your database on local disk or a SAN (Storage Area Network)? If you company has a storage team, get to know them, and understand what tools they use to monitor [IOPS (Input/output Operations per Second)](http://en.wikipedia.org/wiki/IOPS).

**[](http://www.essentialsql.com/studyplanarticle)**

**How to read a query plan**

As a junior DBA you should know how to generate and read a basic query plan.  I wouldn’t expect you know fully understand all verbiage, but several key phrases, such as “Full Table Scan,” and “Nested Loops” should jump out as red flags. Also, you should understand, when the optimizer recommends a change be made, why that change would work, and what the trade-offs are in making it.  For instance, the optimizer may recommend that an index be created.  It this to compensate for poor programming?  Also, if you add it, could something else suffer, such as a process to insert or update data?

**Knowledge of normalization**

Normalized tables are the foundation of a well-designed relational database, yet they can also be its bane. A junior DBA should understand and know how to put data into [1](http://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/" \o "Get Ready to Learn SQL: 8. Database Normalization Explained in Simple English)[st](http://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/" \o "Get Ready to Learn SQL: 8. Database Normalization Explained in Simple English)[, 2](http://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/" \o "Get Ready to Learn SQL: 8. Database Normalization Explained in Simple English)[nd](http://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/" \o "Get Ready to Learn SQL: 8. Database Normalization Explained in Simple English)[, and 3](http://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/" \o "Get Ready to Learn SQL: 8. Database Normalization Explained in Simple English)[rd](http://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/" \o "Get Ready to Learn SQL: 8. Database Normalization Explained in Simple English)[normal form](http://www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english/" \o "Get Ready to Learn SQL: 8. Database Normalization Explained in Simple English).  Why is normalization important and when can it become a liability? Knowing the difference between a [Primary, Foreign, and Unique key is important](http://www.essentialsql.com/get-ready-to-learn-sql-7-simplified-data-modeling/" \o "Get Ready to Learn SQL: 7. Simplified Data Modeling).  Also is understanding and knowing how to enforce one-to-one, one-to-many relationships.

**Knowledge of SQL**

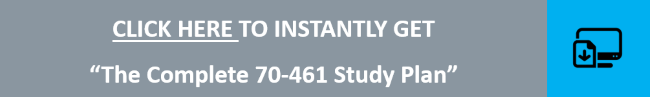
It might seem obvious, but a DBA should have a really good grasp of both SQL DML (Data Manipulation Language) and DDL (Data Definition Language). DML covers items such as Select, Update, Insert and Delete.  You should understand all the major clauses such as WHERE, GROUP BY, HAVING, and ORDER BY.  In addition you should be comfortable with sub queries and[joins](http://www.essentialsql.com/get-ready-to-learn-sql-12-introduction-to-database-joins). DDL covers items such as CREATE TABLE and ALTER TABLE.  A junior DBA should understand how to create and modify tables and indexes and [know the difference between deleting records, truncating a table, or dropping it!](http://www.essentialsql.com/what-is-the-difference-between-truncate-and-delete-in-sql-server/" \o "What is the Difference between Truncate and Delete in SQL Server?) And… don’t forget views!

**Operating System**

As a DBA you need to be familiar with the OS (Operating System) your DBMS lives within.  You should understand nuances of your operating system, such as security settings, integration with Active Directory, LDAP (Lightweight Directory Access Protocol), and naming conventions.  Also, how is your DBMS started?  What scripts are used to start, shutdown, or lock users out of your database?

**Scripting**

To become efficient it is important to know when OS scripting, such as PowerScript, can help you manage your servers.  Consider having ten or more DB servers?  If you had to shut down all the DBMS on them, would you individually log in and manually shut them down or use a script? In my book, if you want to get an edge on other junior DBA’s you learn scripting.  This will only make you more efficient at your job.

**[](http://www.essentialsql.com/studyplanarticle)**

**Stored Procedures and Triggers**

I made [stored procedures](http://www.essentialsql.com/what-is-a-stored-procedure/" \o "What is a Stored Procedure?) and [triggers](http://www.essentialsql.com/what-is-a-database-trigger/" \o "What is a Database Trigger?" \t "_blank) a separate category since they’re more like programming than pumping out SQL statements.  Regardless, you should know when to use them and recommend to your developers their virtues versus writing SQL in code. Also, many third party applications ship with [Stored Procedures](https://www.essentialsql.com/introduction-stored-procedures/) and Triggers.  Being able to read them and understand their logic can go a long ways when trying to troubleshoot a performance issue. The sooner you can avoid treating them like black boxes, the quicker you’ll understand and become more comfortable with the databases you support.