Accessing Your Database

UGA's web standard hosting service has multiple servers for different users. You will only ever directly login to a single server to manage content and databases. We'll call it the *management node*. Your content and applications are served to end users from the *web server nodes*.

By logging into the management node using SSH, you will have full control of your databases. For a given database, there are actually three databases: one for development, stage, and production. To communicate with the proper database from the management node, you must specify the fully qualified domain name of the database server, such as `webhostingdb-c1n0.cc.uga.edu`.

In your application code, served by the web server nodes, you need only use the basic host name of `webhostingdb-c1n${node_number}`, where `${node_number}` is the particular clustered database node (typically 0). That is to say that **most web applications will simply use `webhostingdb-c1n0` as the database host**, without adding `.cc.uga.edu` or anything after it. This works because the web servers are in the same DNS domain as their corresponding database servers. This way, changing the database server used by the application as it is moved from one stage of development to another (i.e. stage to production) is not necessary. The code will automatically connect to the database which corresponds to the current stage of development. **NOTE:** By design, you **cannot** connect to a UGA hosting database server outside of the web server node's environment, i.e. your code cannot connect to the dev environment database server from the production web environment.

**List of Database Server Names**

- `webhostingdb-c1n0.dev.uga.edu` - development
- `webhostingdb-c1n0.stage.uga.edu` - staging
- `webhostingdb-c1n0.cc.uga.edu` - production

**Connecting From a Third-Party MySQL Client**

Of course, since SSH is supported, you may choose to administer your databases with third-party software, such as "MySQL Query Browser". To do so, simply use a port forward to the desired database server when connecting to the management node (`ugawebhosting.uga.edu`). With a standard, terminal-based SSH client, this is done by using the "-L" switch, as in:

- `ssh -L 3306:webhostingdb-c1n0.dev.uga.edu:3306 myusername@ugawebhosting.uga.edu`

In the third-party software, specify `localhost` as the database server. Communication from your client to the management node will be encrypted and then forwarded to the designated database server.

**Database Security**

Be sure to protect your database credentials. In your code, consider placing credentials in include files above the document root (`public_html`), possibly in the `etc` directory. You should be able to remove all group and other user permissions for this `include` file (600 / `rw-------`).

Database servers can only be accessed from the management node, to which one's SSH sessions connect, and from the UGA Webhosting web servers which correspond to it's domain. For example, the staging web server is the only web server which can connect to the staging database server. In an SSH session, all database servers are accessible.

**Redundancy**

The production database is configured with a multi-master replica. This means that you can point your code at either `webhostingdb-c1n0` or `webhostingdb-c1n1`. The data should replicate bidirectionally very quickly. If you are writing custom code, you might consider balancing requests...
Support

We do not offer support for this feature. Our backups are for disaster-recovery purposes only.

Command-Line Operations

From the terminal (SSH), you can do anything to your database, from creating tables to backing up and migrating the database between development stages.

Here are some examples (using the uw_ballpark user as an example):

Backing Up A Database

Here's an example of how to backup the uw_ballpark database in the development environment.

- `mysqldump -h webhostingdb-c1n0.dev.uga.edu -u uw_ballpark -p uw_ballpark > /usr/www/dev/ballpark/other/uw_ballpark.sql`

Migrating A Development Database to Staging

One can use a backup of the development database to populate the staging database. This simple example completely obliterates any existing content in the staging database:

- `mysql -h webhostingdb-c1n0.stage.uga.edu -u uw_ballpark -p uw_ballpark < /usr/www/dev/ballpark/other/uw_ballpark.sql`

Getting to a MySQL Session Prompt

To issue raw SQL statements to MySQL for the uw_ballpark, do the following:

- `mysql -h webhostingdb-c1n0.dev.uga.edu -u uw_ballpark -p uw_ballpark`

Connecting to 3rd-Party Databases

If you have a database hosted elsewhere, you may connect to it using the MySQL client installed on the web servers. You will likely need to reference the list of web servers to create users and firewall rules in your third-party database.