

Useful SQL

A collection of SQL queries that might be useful in different situations.

- [Extract list of users and groups](#)
- [Moving database from live to test](#)

Extract list of users and groups

One line for each group

```
SELECT user.Name, user.EMAIL, user.IsActive, usergroup.Groupname, usergroup.LDAPNAME
FROM user
INNER JOIN usergroupmember ON usergroupmember.UserID = user.UserID
INNER JOIN usergroup ON usergroupmember.GroupID = usergroup.GroupID
ORDER BY 1
```

One line for each user

```
SELECT user.Name, user.EMAIL, user.IsActive, GROUP_CONCAT(usergroup.Groupname) AS ListOfGroups
FROM user
INNER JOIN usergroupmember ON usergroupmember.UserID = user.UserID
INNER JOIN usergroup ON usergroupmember.GroupID = usergroup.GroupID
GROUP BY usergroup.GroupID
ORDER BY 1
```

Moving database from live to test

Option: Database creation

If there is no database already please perform the following steps (assuming new db is named "sandbox")

```
mysql -uUSERNAME -pPASSWORD
```

```
CREATE DATABASE IF NOT EXISTS sandboxbase;  
CREATE DATABASE IF NOT EXISTS sandboxtest;  
CREATE DATABASE IF NOT EXISTS sandboxlive;  
CREATE USER 'sandboxroot'@'localhost' IDENTIFIED BY 'TempusServaFTW!';  
GRANT ALL PRIVILEGES ON sandboxlive.* TO 'sandboxroot'@'localhost';  
GRANT ALL PRIVILEGES ON sandboxtest.* TO 'sandboxroot'@'localhost';  
GRANT ALL PRIVILEGES ON sandboxbase.* TO 'sandboxroot'@'localhost';  
FLUSH PRIVILEGES;
```

Database transfer

Choose ONE of the options below that best suits your requirements.

Database transfer: Different MySQL server / Same schema name

Export data from LIVE server

```
mysqldump -uUSERNAME -pPASSWORD --databases tsbase tslive tstest > dump.sql
```

Import data to TEST server

```
mysql -u USERNAME -pPASSWORD --force < dump.sql
```

If port 3306 is open you might consider network streaming and using pipes. Note that a remote user is needed on the other server.

```
mysqldump ... | mysql ... -h SERVERNAME
```

```
mysqldump -uLOCAL_USR -pLOCAL_PWD --databases tsbase tslive ttest | mysql -u REMOTE_USR -  
pREMOTE_PWD -h SERVERNAME --force
```

Database transfer: Same MySQL server / Different schema name

Dumping data

Export data from LIVE server

```
mysqldump -uUSERNAME -pPASSWORD tsbase > dumpBase.sql  
mysqldump -uUSERNAME -pPASSWORD tslive > dumpLive.sql  
mysqldump -uUSERNAME -pPASSWORD ttest > dumpTest.sql
```

For production servers you can avoid table locking using if consistency is not a huge issue (data can be changed and loose internal integrity)

```
mysqldump --skip-lock-tables ...
```

If you are handling BLOB data larger than 24mB consider increasing the buffer

```
mysqldump --max-allowed-packet=1G ...
```

Importing data

Import data to TEST server (assuming name is 'sandbox')

```
mysql -u USERNAME -pPASSWORD sandboxbase < dumpBase.sql  
mysql -u USERNAME -pPASSWORD sandboxlive < dumpLive.sql  
mysql -u USERNAME -pPASSWORD sandboxtest < dumpTest.sql
```

Errors during import

Using MySQL 5.6 or higher you might get an error like this

```
Error code: 1118 Row size too large (> 8126).
```

Resolve by setting this configuration in the MySQL configuration file (see below)

```
innodb_strict_mode = 0
```

Errors during import

Using MySQL 5.6 or higher you might get an error like this

```
ERROR 1067 (42000) at line ... : Invalid default value for ...
```

Resolve by setting this configuration in the MySQL configuration file (see below)

```
sql_mode=STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION
```

Errors during import 2

Using MySQL 5.6 or higher you might get syntax errors like

```
"(0) NULL DEFAULT NULL"
```

Remove the incorrect syntax in dump file

```
sed -i 's/(0) / /g' tslive.sql
```

Editing the MySQL configuration file

Conf file may vary but is often found at

```
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

Export and importing in one go

It is possible to pipe the output from dump directly to the mysql import

```
mysqldump ... | mysql ...
```

For test and live on same machine the full live to test overwrite is

```
mysqldump --skip-lock-tables --max-allowed-packet=1G -uUSERNAME -pPASSWORD tsbase | mysql -u
USERNAME -pPASSWORD sandboxbase
mysqldump --skip-lock-tables --max-allowed-packet=1G -uUSERNAME -pPASSWORD tslive | mysql -u USERNAME
-pPASSWORD sandboxlive
mysqldump --skip-lock-tables --max-allowed-packet=1G -uUSERNAME -pPASSWORD tstest | mysql -u USERNAME
-pPASSWORD sandboxtest
```

Partial transfers without files

In some situations we want to move data around without attached files. Example usage: Frequent overwrite of test environment with data from live.

Windows partial transfer

Assuming that password for the root account is 'TempusServa' and the scheme is tsXXXX the dump commands are.

```
mysqldump --skip-lock-tables -q -Q -u root -pTempusServa tsbase > c:\temp\tsbase.sql
```

```
for /F %A in ('mysql -u root -pTempusServa tslive -Bse "SELECT table_name FROM information_schema.tables
WHERE table_schema = 'tslive' AND table_name NOT LIKE '%_file'"') DO @Echo Dumping %A & CALL mysqldump
--skip-lock-tables -q -Q -u root -pTempusServa tslive %A >> c:\temp\tslive.sql
```

```
for /F %A in ('mysql -u root -pTempusServa tstest -Bse "SELECT table_name FROM information_schema.tables
WHERE table_schema = 'tstest' AND table_name NOT LIKE '%_file'"') DO @Echo Dumping %A & CALL
mysqldump --skip-lock-tables -q -Q -u root -pTempusServa tstest %A >> c:\temp\tstest.sql
```

For even faster transfers consider excluding the accesslogs too using

```
for /F %A in ('mysql -u root -pTempusServa tslive -Bse "SELECT table_name FROM information_schema.tables
WHERE table_schema = 'tslive' AND table_name NOT LIKE '%_file' AND table_name NOT LIKE '%_accesslog'"')
DO @Echo Dumping %A & CALL mysqldump --skip-lock-tables -q -Q -u root -pTempusServa tslive %A >>
c:\temp\tslive.sql
```

Configuration changes

Important information: The following script will update the database **tslive**, for the sandbox copy example please **USE sandboxlive**;

Run the following commands **before** you start the server

```
mysql -uUSERNAME -pPASSWORD
```

```
USE tslive;
UPDATE systempolicy SET PolicyValue = 'false' WHERE PolicyName='serviceAutostart';
UPDATE systempolicy SET PolicyValue = 'true' WHERE PolicyName='smtpTestMode';
UPDATE systempolicy SET PolicyValue = 'localhost' WHERE PolicyName='smtpServer';
UPDATE systempolicy SET PolicyValue = 'localhost' WHERE PolicyName='applicationServer';
UPDATE systempolicy SET PolicyValue = 'http://localhost' WHERE PolicyName='applicationURL';
UPDATE systempolicy SET PolicyValue = WHERE PolicyName LIKE 'folder%';
UPDATE systempolicy SET PolicyValue = 'C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps'
WHERE PolicyName='applicationBasePath';
```

Note: The following changes can be replaced by setting the values in the application deployment descriptor, starting from version 2500

Rebuild views

Ensure that views exist by rebuilding them

```
Designer > Modules > Admin services > RebuildViews
```

Troubleshooting

Server not started

- Check application server log files for information
- Tomcat
 - Check xml descriptor is found in [TOMCAT]/conf/Catalina/localhost
 - Check war file is unpacked correctly in [TOMCAT]/webapps

Cannot select fieldtype when adding or editing fields

1. Run the following SQL command

```
USE tsbase;
DROP VIEW IF EXISTS viewdatatypeselector;
CREATE VIEW viewdatatypeselector AS
SELECT
    systemdatatype.FeltTypeID AS `FeltTypeID`,
    concat(systemdatatypepackage.TypePrefix,': ',systemdatatype.FeltType) AS `FeltType`
FROM systemdatatype
JOIN systemdatatypepackage
    ON systemdatatype.PackageID = systemdatatypepackage.PackageID
WHERE systemdatatype.IsSelectable = 1)
ORDER BY systemdatatypepackage.SortOrder, systemdatatype.FeltType;
```

1. Repeat the "Rebuild views above"

Moving parts of an existing installation (application, no data)

Moving part of solution

- Create a new database
- Copy all tables starting with "form"
- Remove all unwanted entries
- Dump the full database
- Restore database on target system

After deleting records you might want to clean up the foreign keys constraints

```
DELETE FROM formconfig WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formfield WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formfieldblock WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formfieldcategory WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formfieldlookup WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM forminterface WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formpage WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formpermission WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formstatus WHERE SagID NOT IN (SELECT SagID FROM form);
DELETE FROM formstatusaction WHERE SagID NOT IN (SELECT SagID FROM form);
```

```
DELETE FROM formstatusflow WHERE SagID NOT IN (SELECT SagID FROM form);
```