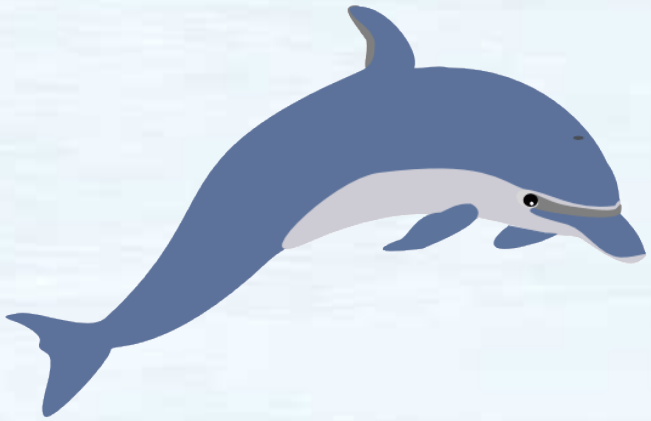
MySQL



CC ICT-SUD

Installation and configuration

Package installation

- Some distribution split the client and server part into separate packages
- Distribution-depedent
  - Arch Linux: # pacman -Sy mysql
  - Ubuntu: $ sudo apt-get install mysql-server
  - Fedora: # yum install mysql mysql-server

Run MySQL daemon

- General commands:

  cd /usr; /usr/bin/mysqld_safe &

  install support-files/mysql.server into your boot system
- Arch Linux: /etc/rc.d mysqld start
- Ubuntu: sudo invoke-rc.d mysql start
- Fedora: service mysqld start
- Or just reboot

Secure installation

- Run this helper script:

  # /usr/bin/mysql_secure_installation

  - Set a MySQL root password (do not leave it empty!)
  - Remove anonymous users
  - Disallow root login remotely
  - Remove test database and access to it
  - Reload privilege tables now

- Manual:

  /usr/bin/mysqladmin -u root password 'new-password'
  /usr/bin/mysqladmin -u root -h cvs password 'new-password'

Testing and access

- cd /usr/mysql-test ; perl mysql-test-run.pl
- Local access using MySQL's command line tool:

  $ mysql -p -u root

- For remote access on the TCP port 3306, edit /etc/mysql/my.cnf and comment out:

  skip-networking

  also enable access through TCP Wrapper in /etc/hosts.allow:

  mysqld: ALL : ALLOW
  mysqld-max: ALL : ALLOW

Adding user accounts

- Example: a safe account for a certain local site
- For best security make one account per site or application with different passwords. E.g.:

```
mysql> CREATE USER 'mysite'@'localhost'
IDENTIFIED BY 'mypassword';

mysql> GRANT SELECT, INSERT, UPDATE,
DELETE ON mysitedb.* TO 'mysite'@'localhost';

mysql> FLUSH PRIVILEGES;
```

Integer type hinter

- [0..255]: TINYINT UNSIGNED (1 byte)
- [-128..127]: TINYINT (1 byte)
- [0..65535]: SMALLINT UNSIGNED (2 bytes)
- [-32768..32767]: SMALLINT (2 bytes)
- [0..16777215]: MEDIUMINT UNSIGNED (3 bytes)
- [-8388608..8388607]: MEDIUMINT (3 bytes)
- [0..4294967295]: INT UNSIGNED (4 bytes)
- [-2147483648..2147483647]: INT (4 bytes)
- [0..18446744073709551615]: BIGINT UNSIGNED (8 bytes)
- [-9223372036854775808, +9223372036854775807]: BIGINT

String type hinter (1)

- Maxlen<=3:
  - rtrim=NULL:
    - storage=NULL: CHAR
    - storage=fixed: CHAR
    - storage=variable: VARCHAR
    - storage=offpage: TINYTEXT
  - rtrim=FALSE:
    - storage=NULL: VARCHAR
    - storage=fixed: N.A.
    - storage=variable: VARCHAR
    - storage=offpage: TINYTEXT
  - rtrim=TRUE:
    - storage=NULL: CHAR
    - storage=fixed: CHAR
    - storage=variable: VARCHAR
    - storage=offpage: TINYTEXT

String type hinter (2)

- Maxlen<=255:
  - rtrim=NULL:
    - storage=NULL: VARCHAR
    - storage=fixed: CHAR
    - storage=variable: VARCHAR
    - storage=offpage: TINYTEXT
  - rtrim=FALSE:
    - storage=NULL: VARCHAR
    - storage=fixed: N.A.
    - storage=variable: VARCHAR
    - storage=offpage: TINYTEXT
  - rtrim=TRUE:
    - storage=NULL: VARCHAR
    - storage=fixed: CHAR
    - storage=variable: VARCHAR
    - storage=offpage: TINYTEXT

String type hinter (3)

- Maxlen<=65535:
  - rtrim=NULL:
    - storage=NULL: VARCHAR
    - storage=fixed: N.A.
    - storage=variable: VARCHAR
    - storage=offpage: TEXT
  - rtrim=FALSE:
    - storage=NULL: VARCHAR
    - storage=fixed: N.A.
    - storage=variable: VARCHAR
    - storage=offpage: TEXT
  - rtrim=TRUE:
    - storage=NULL: VARCHAR
    - storage=fixed: N.A.
    - storage=variable: VARCHAR
    - storage=offpage: TEXT

String type hinter (4)

- Maxlen<=16777215:
  - rtrim=NULL:
    - storage=NULL: MEDIUMTEXT
    - storage=fixed: N.A.
    - storage=variable: N.A.
    - storage=offpage: MEDIUMTEXT
  - rtrim=FALSE:
    - storage=NULL: MEDIUMTEXT
    - storage=fixed: N.A.
    - storage=variable: N.A.
    - storage=offpage: MEDIUMTEXT
  - rtrim=TRUE:
    - storage=NULL: MEDIUMTEXT
    - storage=fixed: N.A.
    - storage=variable: N.A.
    - storage=offpage: MEDIUMTEXT

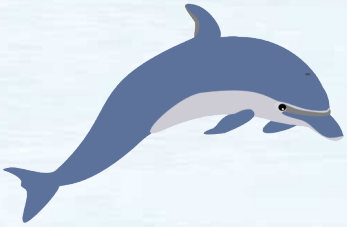
String type hinter (5)

- Maxlen<=4294967295:
  - rtrim=NULL:
    - storage=NULL: LONGTEXT
    - storage=fixed: N.A.
    - storage=variable: N.A.
    - storage=offpage: LONGTEXT
  - rtrim=FALSE:
    - storage=NULL: LONGTEXT
    - storage=fixed: N.A.
    - storage=variable: N.A.
    - storage=offpage: LONGTEXT
  - rtrim=TRUE:
    - storage=NULL: LONGTEXT
    - storage=fixed: N.A.
    - storage=variable: N.A.
    - storage=offpage: LONGTEXT

Some MySQL table types

- MyISAM (default): great performance, nontransactional, table-level locking, full-text search indexes, up to 256 TB
- InnoDB: transaction-safe, foreign keys, clustered indexes, row-level locking, up to 64 TB
- Memory: temporary, extremely fast, hash indexes by default
- How to choose: CREATE TABLE … (…) ENGINE=INNODB;

MySQL



CC ICT-SUD

Installation and configuration

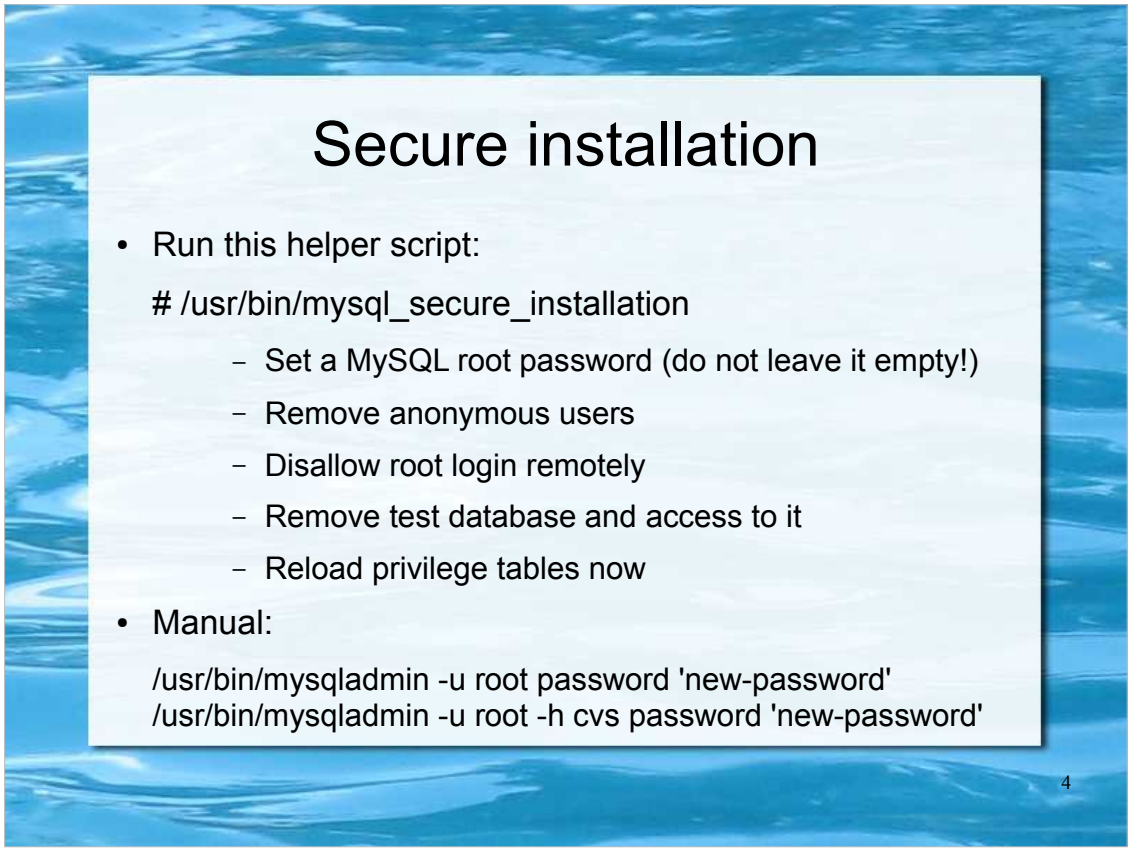
Package installation

- Some distribution split the client and server part into separate packages
- Distribution-depedent
  - Arch Linux: # pacman -Sy mysql
  - Ubuntu: $ sudo apt-get install mysql-server
  - Fedora: # yum install mysql mysql-server

MySQL safe mode adds the following safety features:

- Restarts the server when an error occurs
- Logs runtime information to an error log file

Datadir is /var/lib/mysql/DATABASE_NAME. You can make a database backup by copying these files. To get a consistent backup first stop the server or lock tables to backup (see MySQL manual for details).



Setting the root password ensures that nobody can log into the MySQL root user without the proper authorisation.

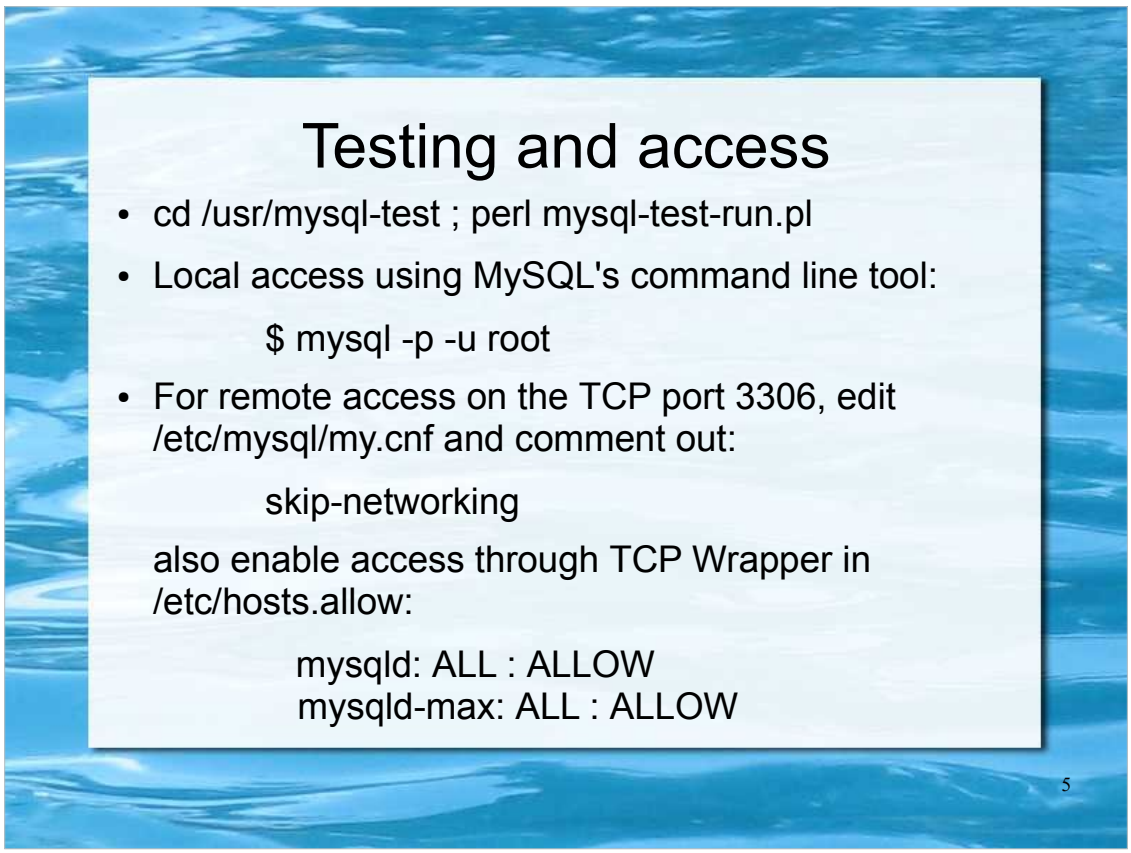
By default, a MySQL installation has an anonymous user, allowing anyone to log into MySQL without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network. The root account is used to maintain your MySQL users and databases.

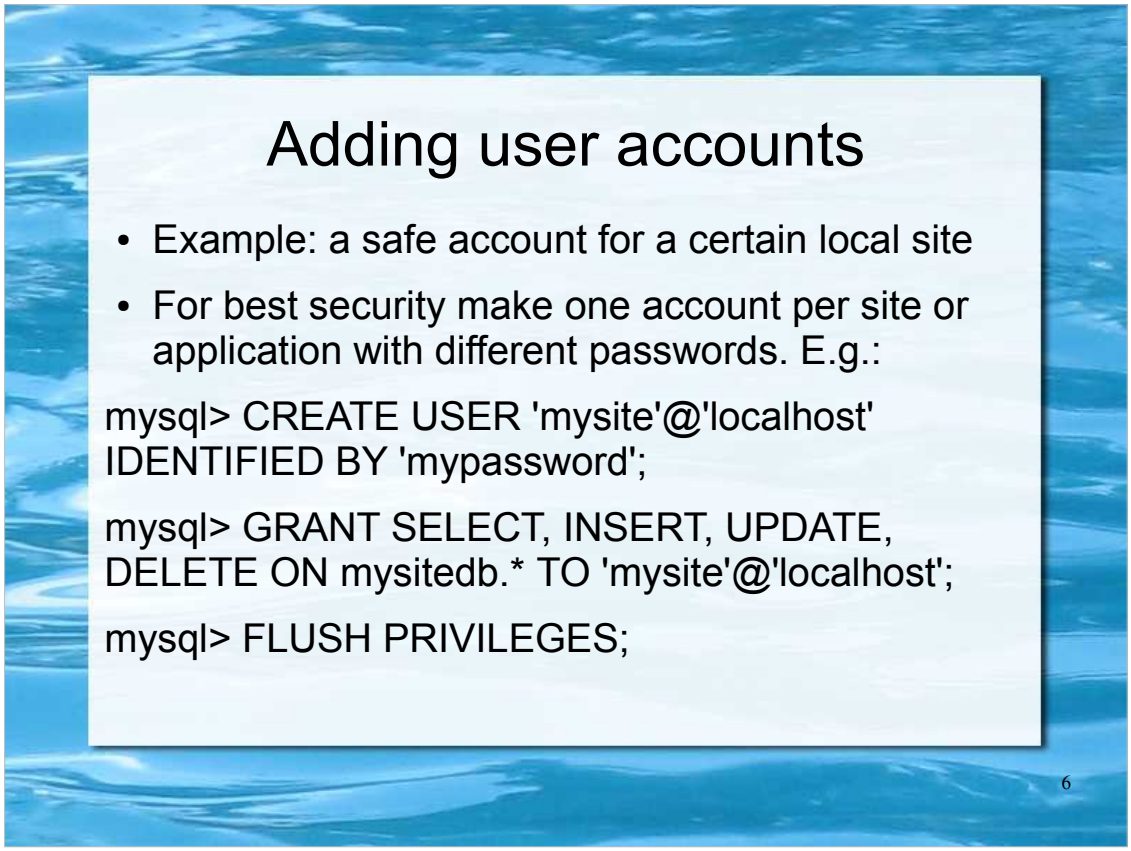
By default, MySQL comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Why two commands? If there is a second entry for root and your hostname is left with no password set, everybody from your host probably could gain full access.



The examples above are the simplest case, telling tcpd to allow connections from anywhere. You may wish to use a more-appropriate choice of permissible sources instead of 'ALL'. Just make sure that localhost and the IP address (numeric or DNS) of the interface by which you connect are specified.



Database privileges are stored in a relational database named “mysql”. To create accounts we could manipulate this db directly, but the preferred method is to use account-creation statements, such as CREATE USER or GRANT.

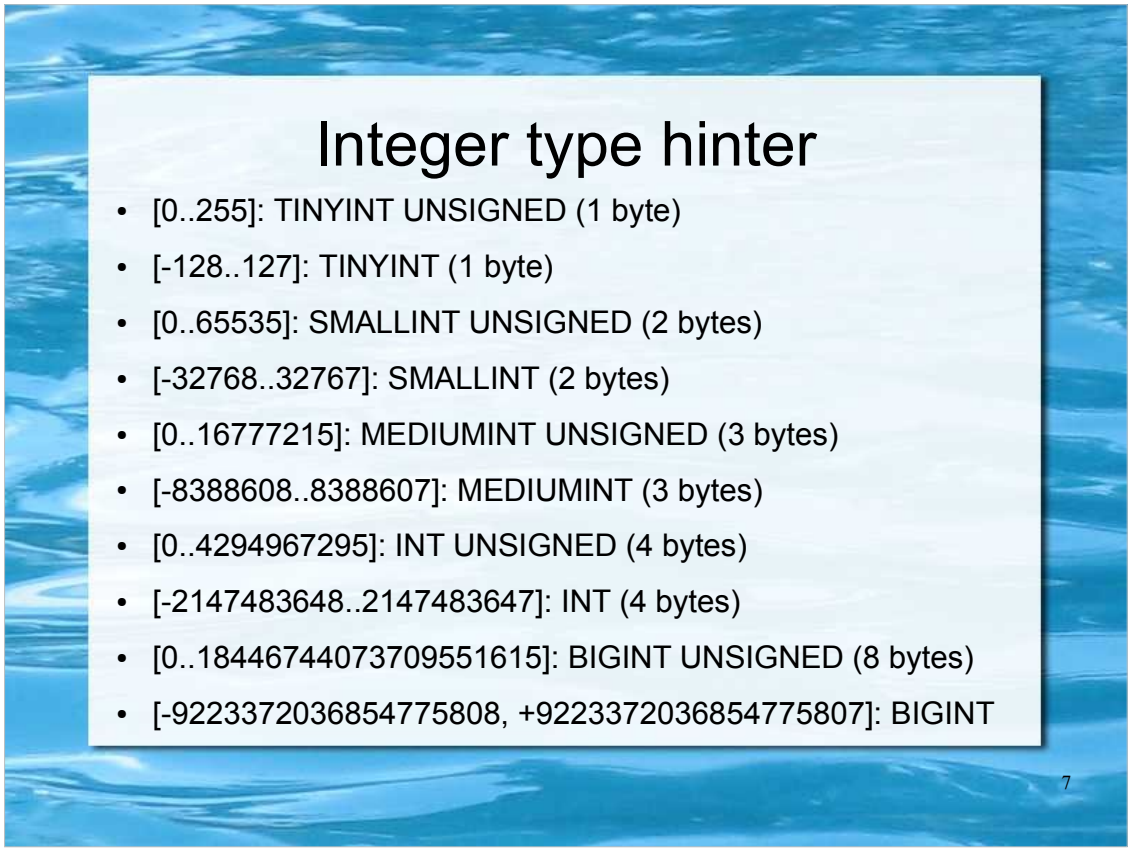
First connect to the MySQL server as the MySQL root user as described before: $ mysql -uroot -p

Use the '%' wildcard for the host part, if you want the user to be able to connect from any host.

To make the new accounta available you must tell the server to reload the grant tables at the end using FLUSH PRIVILEGES.

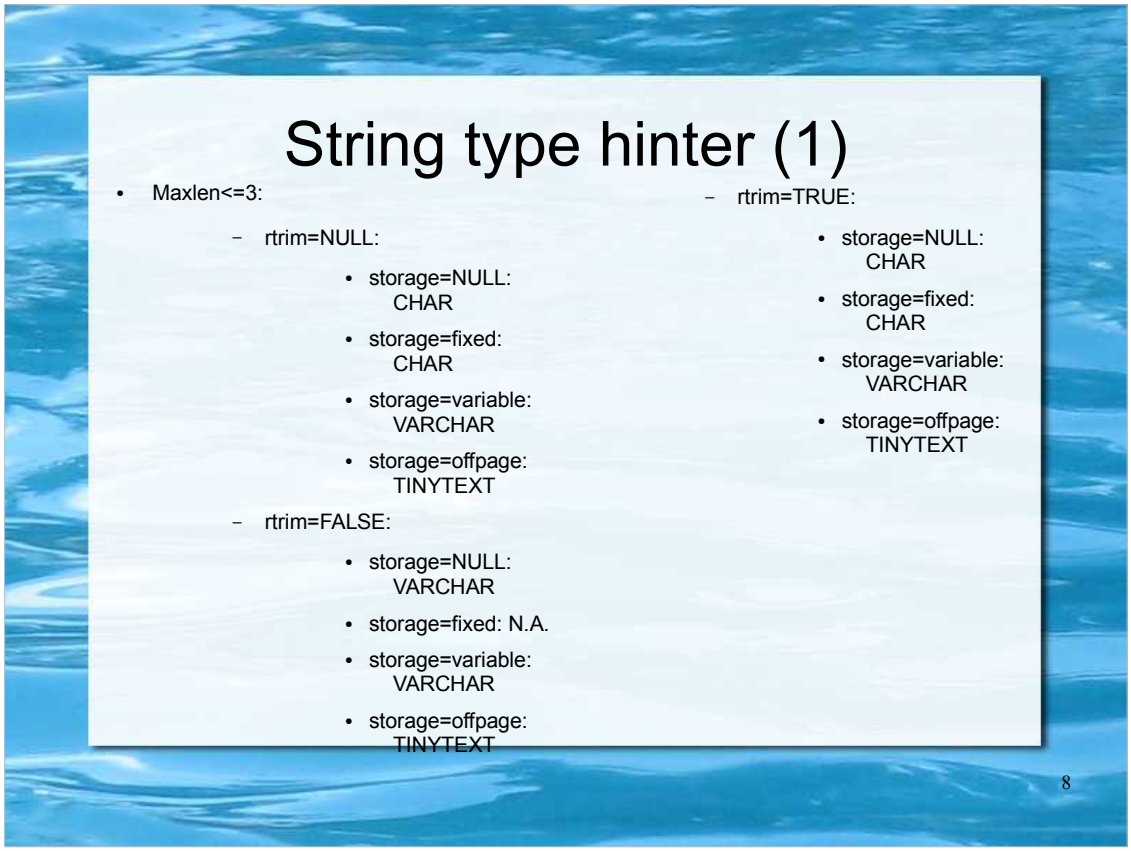
It is a good security practice to grant all and only the privileges that are strictly necessary. E.g. usually a web site won't need create and drop privileges. You can create the database from a schema dump by logging as root once for all: $ mysql -uroot -p <schema.sql

For more examples and options see the MySQL manual.



Select the best MySQL integer type to use based on range constraints. This minimizes the amount of storage required.

Note that the integer range results from the two complements binary representation method.

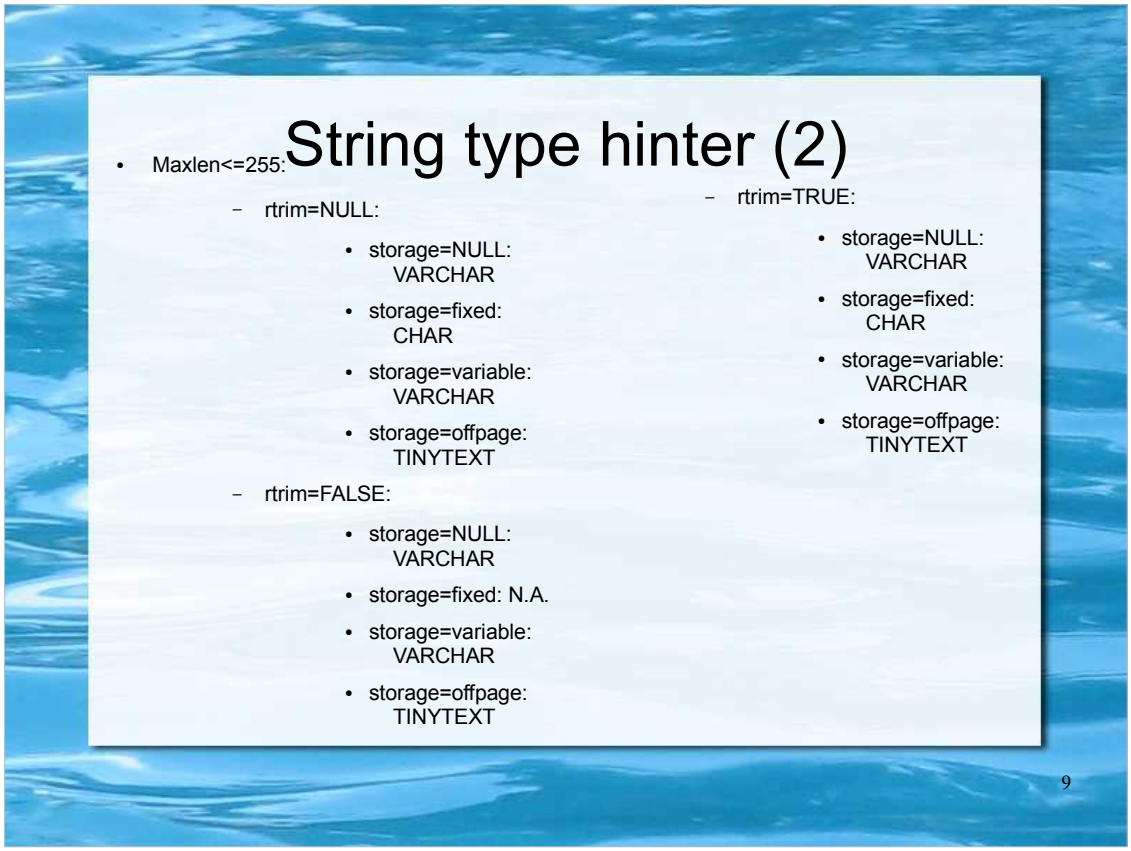


Select between different binary and string data types based on these three attributes (NULL means "don't care"):

- maxlen: maximum number of characters or bytes (for binary data) to store
- rtrim: whether trailing spaces are removed or retained
- storage: where and how the column data are stored

This decision tree will allow you to make a good choice. If you don't use it and you guess wrong MySQL will apply some silent column changes, unless strict SQL mode is enabled.

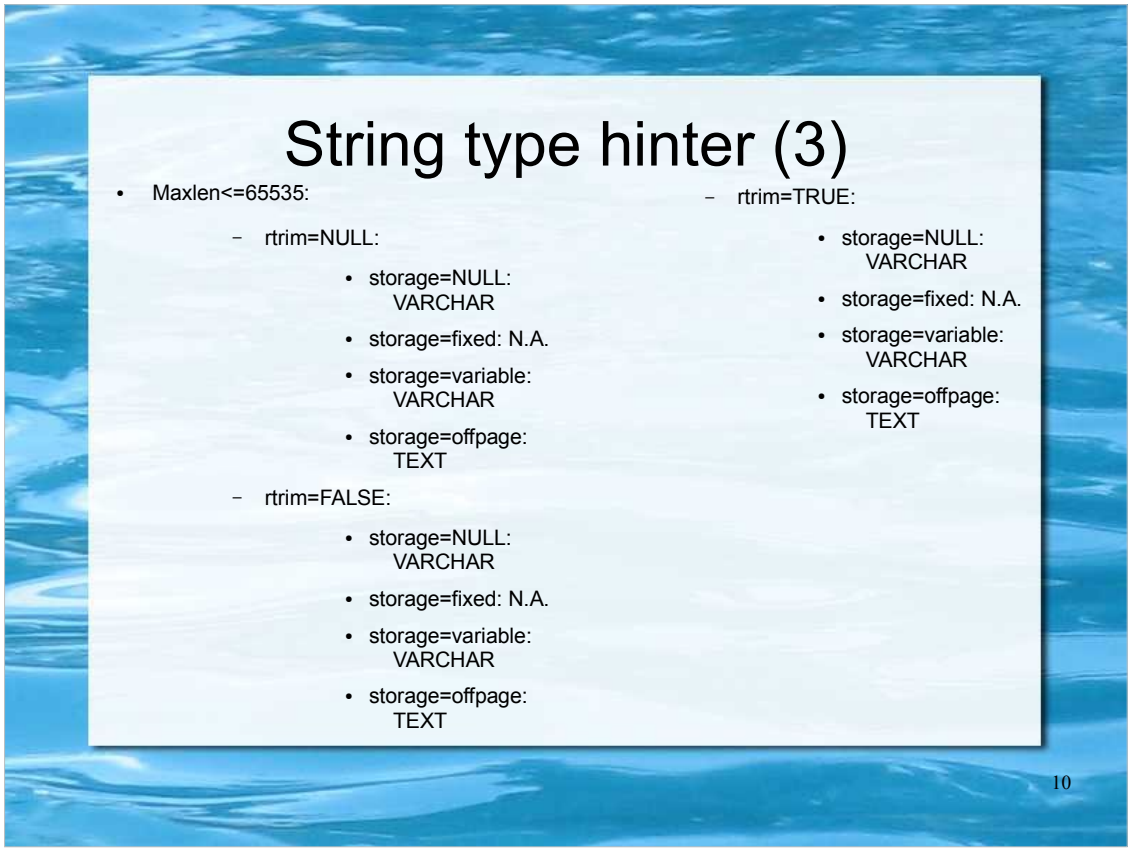
N.A. = not available



Select between different binary and string data types based on these three attributes (NULL means "don't care"):

- maxlen: maximum number of characters or bytes (for binary data) to store
- rtrim: whether trailing spaces are removed or retained
- storage: where and how the column data are stored

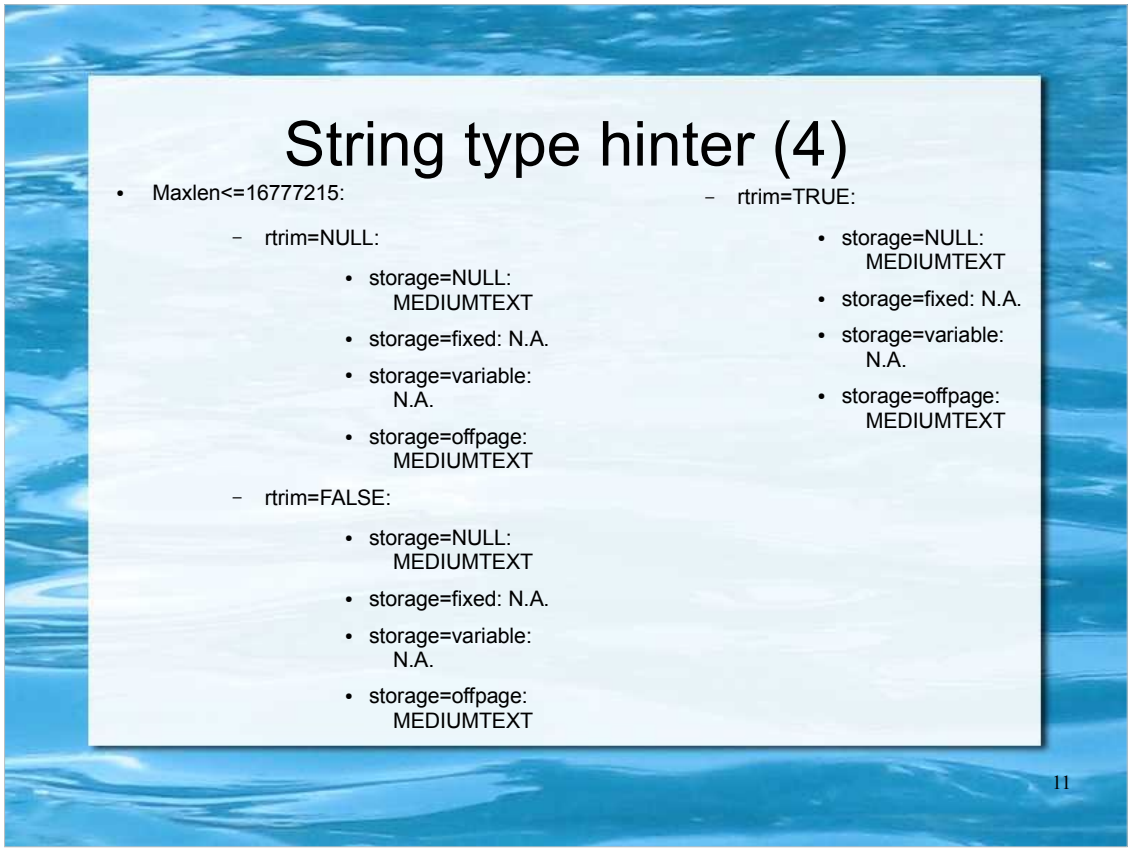
This decision tree will allow you to make a good choice. If you don't use it and you guess wrong MySQL will apply some silent column changes, unless strict SQL mode is enabled.



Select between different binary and string data types based on these three attributes (NULL means "don't care"):

- maxlen: maximum number of characters or bytes (for binary data) to store
- rtrim: whether trailing spaces are removed or retained
- storage: where and how the column data are stored

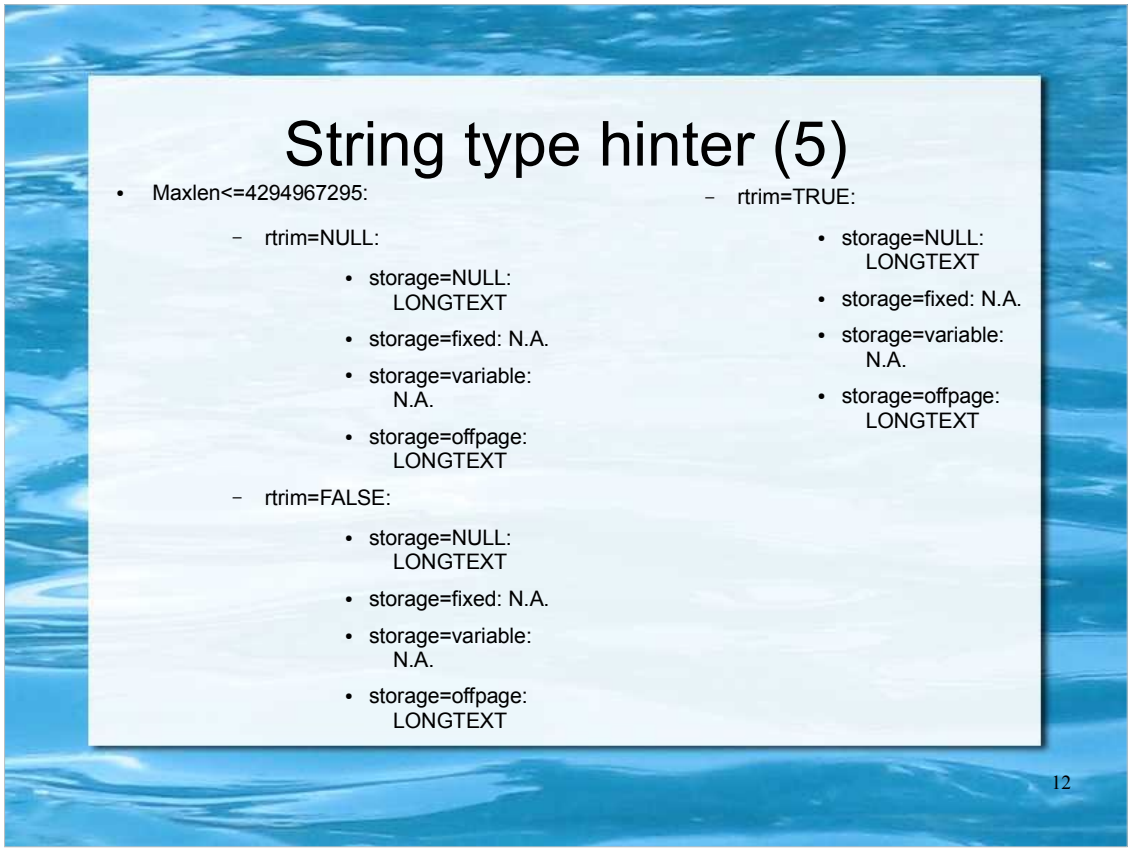
This decision tree will allow you to make a good choice. If you don't use it and you guess wrong MySQL will apply some silent column changes, unless strict SQL mode is enabled.



Select between different binary and string data types based on these three attributes (NULL means "don't care"):

- maxlen: maximum number of characters or bytes (for binary data) to store
- rtrim: whether trailing spaces are removed or retained
- storage: where and how the column data are stored

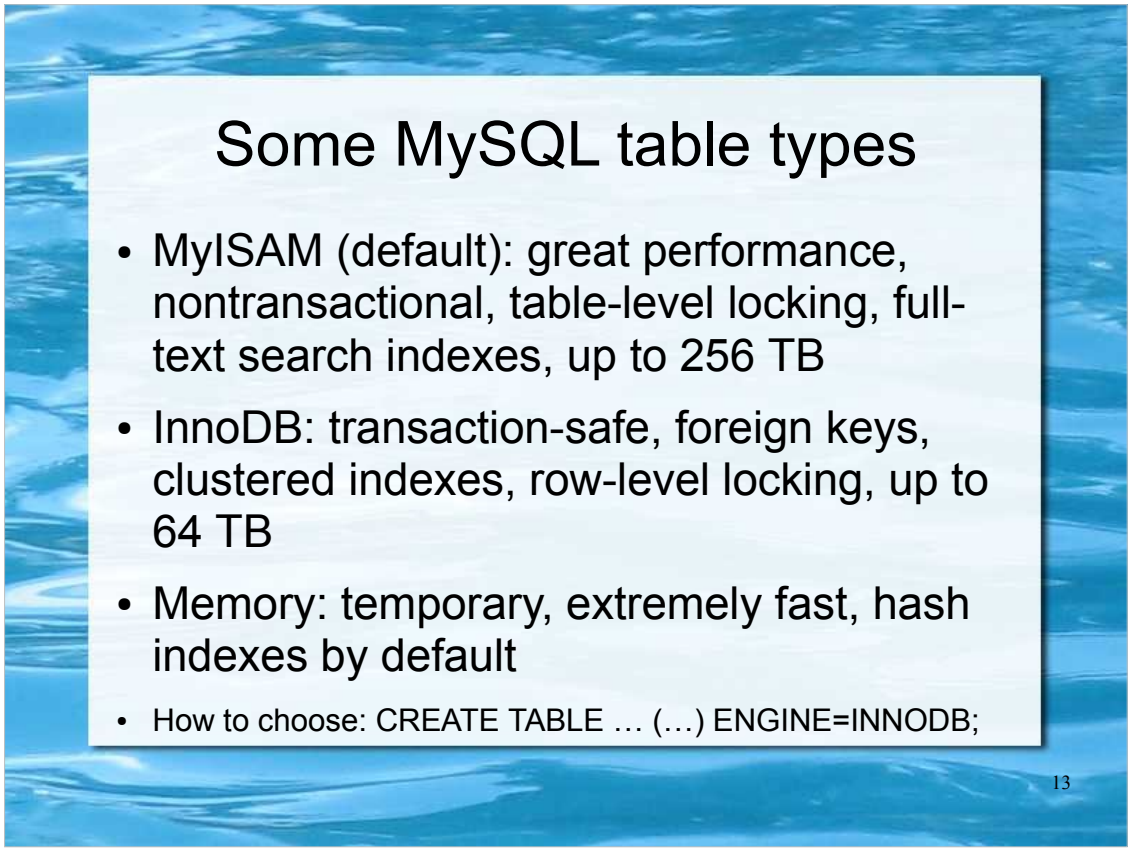
This decision tree will allow you to make a good choice. If you don't use it and you guess wrong MySQL will apply some silent column changes, unless strict SQL mode is enabled.



Select between different binary and string data types based on these three attributes (NULL means "don't care"):

- maxlen: maximum number of characters or bytes (for binary data) to store
- rtrim: whether trailing spaces are removed or retained
- storage: where and how the column data are stored

This decision tree will allow you to make a good choice. If you don't use it and you guess wrong MySQL will apply some silent column changes, unless strict SQL mode is enabled.



Also called storage engines.

MyISAM is used the most in Web, data warehousing and all other read heavy application enviroments /(many reads, some writes). Table-level locking: nobody can access any other record of the same table during an UPDATE.

InnoDB (best for some reads, many writes cases where reability is required) is ACID compliant and has commit, rollback, and crash-recovery capabilities to protect user data. InnoDB stores user data in clustered indexes to reduce I/O for common queries based on primary keys. To maintain data integrity, InnoDB also supports FOREIGN KEY referential-integrity constraints. Row level locking means during an UPDATE, nobody can access that particular row, until the locking transaction issues a COMMIT.

Memory should only be used for temporary tables: if MySQL crashes or it's shutted down you will lose all data stored in them!