MySQL

Adam Montgomery

3/4/19

Data Types

Type	Size
INT	
BIGINT	
SMALLINT	
TINYINT	
MEDIUMINT	
CHAR	0-255
VARCHAR	0-255
TEXT	0-65535
DATE	YYYY-MM-DD
DATETIME	YYYY-MM-DD HH:MM:SS
TIMESTAMP	YYYYMMDDHHMMSS
TIME	HH:MM:SS
FLOAT	to 23 digits
DOUBLE	24-53 digits
DECIMAL	Double as string
BOOLEAN	TINYINT(1)
BLOB	String(0-65535)
ENUM	One of preset
SET	Selection of presets

Creating Tables

Tables are the basic building blocks of databases. They store information in columns with defined data-types. Each row is a record, and should be unique for most tables.

```sql
CREATE TABLE tblName (
rowID INTEGER,
col2 VARCHAR(20),
col3 TIMESTAMP);
```

Deleting Tables

Tables can be deleted or *dropped* from the database when they are no longer needed.

```sql
DROP TABLE tblName;
```

Changing Tables

Table columns can be changed after the table is created. It isn’t a good practice to change a table once it contains records because the existing records will not have any data in their new column/s.

```
ALTER TABLE ADD COLUMN test INT AFTER rowID;
```

Selecting Records

The select statement is probably the most used type of query statement in SQL. It returns rows of table data, and certain columns can be specified or left out.

```
SELECT * FROM Customers;
SELECT * FROM Customers WHERE CustomerID=10;
SELECT firstName, lastName FROM Customers;
```

Inserting Records

The insert statement is used to create rows of data in a table

```
INSERT INTO Customers (firstName, lastName)
VALUES(
’Adam’,
’Montgomery’);
```

Updating Records

Updating is useful when you have data that changes on a regular basis.

```
UPDATE Customers SET firstName = ’Thomas’
WHERE lastName = ’Jefferson’;
```

Delete Records

Records can also be deleted easily with the *DELETE* statement.

```
DELETE FROM Customers WHERE CustomerID = 10;
```

Joining Disparate Tables

Join statements can be used to merge 2 or more tables together. This is extremely useful as database tables are supposed to be segregated or *normalized* so that they house only pertinent data.

Type of join	Output
INNER	join on one column where matching
LEFT	* from left and matching from right
RIGHT	* from right and matching from left
OUTER	matching data from left or right

```
SELECT firstName, lastName, address
FROM Customers INNER JOIN ContactDetails
ON Customers.contactID = ContactDetails.contactID;
```

Sometimes these *JOIN* statements can be complicated. The easiest way to think about it is that the join occurs first, and the select statment is applied to the *joined* tables.