Prepared Statements

```
-- Create
PREPARE stmt FROM
"SELECT * FROM users WHERE age > ?";

-- Use
SET @age=30;
EXECUTE stmt USING @age;
DEALLOCATE PREPARE stmt;
```

Variables

```
-- Global
SELECT @@max_connections;

-- Session
SET @user_var = 'value';
SELECT @user_var;

-- Local (in routines)
DECLARE local_var INT;
SET local_var = local_var + 1;
```

Stored Procedures

```
DELIMITER $$
CREATE PROCEDURE procName(IN par1 INT,OUT par2 BOOL)
BEGIN
  -- body
  SELECT col INTO param2 FROM table WHERE id =
      param1;
END$$
DELIMITER ;

CALL procName(1, @out); -- run
DROP PROCEDURE IF EXISTS proc_name; -- delete
```

IF

```
IF x > 0 THEN
  SET y = x
ELSEIF x = 0 THEN
  SET y = 0;
ELSE
  SET y = -x;
END IF;
```

Loops

```
-- WHILE
WHILE counter <= 10 DO
  SET counter = counter + 1;
END WHILE;

-- REPEAT
REPEAT
  SET n = n + 1;
UNTIL n > 5
END REPEAT;

-- Useful keywords
LEAVE -- "break"
ITERATE -- "continue"
```

Views

```
CREATE VIEW viewName AS
SELECT [...]
```

Transactions

```
START TRANSACTION;
[...]
COMMIT;
```

Triggers

```
DELIMITER $$
CREATE TRIGGER trigger_name
BEFORE/AFTER INSERT/UPDATE/DELETE ON table_name
FOR EACH ROW -- trigger granularity
BEGIN
  -- logic
  IF NEW.columnName = OLD.columnName * 2 THEN
    [...]
END$$
DELIMITER ;
```

User-Defined Functions

```
DELIMITER $$
CREATE FUNCTION func_name(param INT)
RETURNS INT
DETERMINISTIC
BEGIN
  DECLARE result INT;
  SET result = param * param;
  RETURN result;
END$$
DELIMITER ;

SELECT func_name(5);
DROP FUNCTION IF EXISTS func_name;
```

Cursors

```
-- Declare handler and cursor variables
DECLARE cursorName CURSOR FOR SELECT ...;

DECLARE done BOOLEAN DEFAULT FALSE;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done =
    TRUE;

-- Use the cursor
OPEN cursorName;
WHILE done IS FALSE DO
  FETCH cursorName INTO var;
  [ ... ]
END WHILE;
CLOSE cursorName;
```