1. The SELECT query selects what you want to come up in the Result grid
2. The FROM selects where you want the information to come from
3. The WHERE clause selects where in the information you want to come out
4. The GROUP BY groups 2 columns or rows together
5. The CREATE TABLE creates a new tables
6. The ALTER TABLE adds a new column or row
7. The USE Selects a schema
8. The INNER JOIN combines 2 tables having the column or row
9. The UNION combines 2 tables regardless of the table, column or row
10. The INSERT INTO just selects the columns you want to go into regardless of the table
11. The ORDER BY defines them in a certain ORDER
12. The LIKE defines a row by a letter, number or word
13. The PRIMARY KEY is a column within a table
14. The FOREIGN KEY is a column within a table and outside the table
15. The VALUES are the new rows being added to the columns
16. The DROP TABLE deletes a table

USE bob\_ross;

SELECT \* FROM elements\_by\_epsiode;

CREATE TABLE happy\_trees(

Palm\_trees varchar(20)

Pine\_trees varchar(20)

Camphor\_trees varchar(20)

);

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4.The varchar limits the amount of rows

USE bob\_ross;

SELECT \* FROM elements\_by\_epsiode;

CREATE TABLE happy\_trees(

Palm\_trees varchar(20)

Pine\_trees varchar(20)

Camphor\_trees varchar(20)

);

ALTER TABLE happy\_trees ADD coconut\_trees varchar(20)

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4.The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

USE bob\_ross;

SELECT \* FROM elements\_by\_epsiode;

CREATE TABLE happy\_trees(

Palm\_trees varchar(20)

Pine\_trees varchar(20)

Camphor\_trees varchar(20)

);

ALTER TABLE happy\_trees ADD coconut\_trees varchar(20)

INSERT INTO happy\_trees (palm\_trees, pine\_trees, camphor\_trees, coconut\_trees)

VALUES (“Florida”, “New York ”, “Texas”, “California”);

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4.The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

6. The INSERT INTO just selects the columns you want to go into regardless of the table

7.The VALUES are the new rows being added to the columns

USE bob\_ross;

SELECT \* FROM elements\_by\_epsiode;

CREATE TABLE happy\_trees(

Palm\_trees varchar(20)

Pine\_trees varchar(20)

Camphor\_trees varchar(20)

);

ALTER TABLE happy\_trees ADD coconut\_trees varchar(20)

INSERT INTO happy\_trees (palm\_trees, pine\_trees, camphor\_trees, coconut\_trees)

VALUES (“Florida”, “New York ”, “Texas”, “California”);

SELECT \* FROM happy\_trees;

DROP TABLE happy\_trees;

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4.The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

6. The INSERT INTO just selects the columns you want to go into regardless of the table

7.The VALUES are the new rows being added to the columns

8. The DROP TABLE deletes a table

USE bob\_ross;

SELECT \* FROM elements\_by\_epsiode;

CREATE TABLE happy\_trees(

Palm\_trees varchar(20)

Pine\_trees varchar(20)

Camphor\_trees varchar(20)

);

ALTER TABLE happy\_trees ADD coconut\_trees varchar(20)

INSERT INTO happy\_trees (palm\_trees, pine\_trees, camphor\_trees, coconut\_trees)

VALUES (“Florida”, “New York ”, “Texas”, “California”);

SELECT \* FROM happy\_trees;

DROP TABLE happy\_trees;

SELECT \* FROM happy\_trees

UNION

SELECT EPISODE,TITLE,APPLE,FRAME,BARN FROM elements\_by\_episode;

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4.The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

6. The INSERT INTO just selects the columns you want to go into regardless of the table

7.The VALUES are the new rows being added to the columns

8. The DROP TABLE deletes a table

9. The UNION combines 2 tables regardless of the table, column or row

USE bob\_ross;

SELECT \* FROM elements\_by\_epsiode;

CREATE TABLE happy\_trees(

Palm\_trees varchar(20)

Pine\_trees varchar(20)

Camphor\_trees varchar(20)

);

ALTER TABLE happy\_trees ADD coconut\_trees varchar(20)

INSERT INTO happy\_trees (palm\_trees, pine\_trees, camphor\_trees, coconut\_trees)

VALUES (“Florida”, “New York ”, “Texas”, “California”);

SELECT \* FROM happy\_trees;

DROP TABLE happy\_trees;

SELECT \* FROM happy\_trees

UNION

SELECT EPISODE,TITLE,APPLE,FRAME,BARN FROM elements\_by\_episode;

SELECT \* FROM happy\_trees WHERE palm\_trees = “Florida”

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4.The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

6. The INSERT INTO just selects the columns you want to go into regardless of the table

7.The VALUES are the new rows being added to the columns

8. The DROP TABLE deletes a table

9. The UNION combines 2 tables regardless of the table, column or row

10. The equals = sign direct what exact row and column you want to go to

USE bob\_ross;

SELECT \* FROM elements\_by\_epsiode;

CREATE TABLE happy\_trees(

Palm\_trees varchar(20)

Pine\_trees varchar(20)

Camphor\_trees varchar(20)

);

ALTER TABLE happy\_trees ADD coconut\_trees varchar(20)

INSERT INTO happy\_trees (palm\_trees, pine\_trees, camphor\_trees, coconut\_trees)

VALUES (“Florida”, “New York ”, “Texas”, “California”);

SELECT \* FROM happy\_trees;

DROP TABLE happy\_trees;

SELECT \* FROM happy\_trees

UNION

SELECT EPISODE,TITLE,APPLE,FRAME,BARN FROM elements\_by\_episode;

SELECT \* FROM happy\_trees WHERE palm\_trees = “Florida”

SELECT \* FROM happy\_trees WHERE palm\_trees IN (SELECT palm\_trees FROM happy\_trees = “ Nevada”

SELECT \* FROM happy\_trees;

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4. The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

6. The INSERT INTO just selects the columns you want to go into regardless of the table

7. The VALUES are the new rows being added to the columns

8. The DROP TABLE deletes a table

9. The UNION combines 2 tables regardless of the table, column or row

10. The equals = sign directs what exact row and column you want to go to

11. The sub query selects what you want to come first

ALTER TABLE happy\_trees ADD coconut\_trees varchar(20)

INSERT INTO happy\_trees (palm\_trees, pine\_trees, camphor\_trees, coconut\_trees)

VALUES (“Florida”, “New York ”, “Texas”, “California”);

SELECT \* FROM happy\_trees;

DROP TABLE happy\_trees;

SELECT \* FROM happy\_trees

UNION

SELECT EPISODE,TITLE,APPLE,FRAME,BARN FROM elements\_by\_episode;

SELECT \* FROM happy\_trees WHERE palm\_trees = “Florida”

SELECT \* FROM happy\_trees WHERE palm\_trees IN (SELECT palm\_trees FROM happy\_trees = “ Nevada”

SELECT \* FROM happy\_trees;  
SELECT \* FROM elements\_by\_episode WHERE TITLE LIKE “%b%” GROUP BY EPISODE ORDER BY TITLE ASC;

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4. The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

6. The INSERT INTO just selects the columns you want to go into regardless of the table

7. The VALUES are the new rows being added to the columns

8. The DROP TABLE deletes a table

9. The UNION combines 2 tables regardless of the table, column or row

10. The equals = sign directs what exact row and column you want to go to

11. The sub query selects what you want to come first

12. The ASC puts them in alphabetical order

13. The LIKE directs which letter in the variable you want to take it to

SELECT \* FROM happy\_trees WHERE palm\_trees IN (SELEECT palm\_trees FROM happy\_trees WHERE palm\_trees = “Nevada”);

SELECT \* FROM happy\_trees;

CREATE TABLE students(

StudentID int NOT NULL,

StudentName varchar(40),

StudentEmail varchar(40),

PersonID int NOT NULL,

PRIMARY KEY(StudentID),

FOREIGN KEY(PersonID)

REFERENCES Persons(PersonID));

INSERT INTO Persons

VALUES(2, 22, “S02E02”);

1. The USE Selects a schema

2. The SELECT query selects what you want to come up in the Result grid

3. The CREATE TABLE creates a new tables

4. The varchar limits the amount of rows

5. The ALTER TABLE adds a new column or row

6. The INSERT INTO just selects the columns you want to go into regardless of the table

7. The VALUES are the new rows being added to the columns

8. The DROP TABLE deletes a table

9. The UNION combines 2 tables regardless of the table, column or row

10. The equals = sign directs what exact row and column you want to go to

11. The sub query selects what you want to come first

12. The ASC puts them in alphabetical order

13. The LIKE directs which letter in the variable you want to take it to

14. The PRIMARY KEY is a column within a table

15.The FOREIGN KEY is a column within a table and outside the table

16.The REFERENCE gets the column with the same name and combines them together

CREATE TABLE students(

StudentID int NOT NULL,

StudentName varchar(40),

StudentEmail varchar(40),

PersonID int NOT NULL,

PRIMARY KEY(StudentID),

FOREIGN KEY(PersonID)

REFERENCES Persons(PersonID));

INSERT INTO students

VALUES(2, nick2, “nick2@gmail.com”);

1. The SELECT query selects what you want to come up in the Result grid

2. The CREATE TABLE creates a new tables

3. The varchar limits the amount of rows

4. The ALTER TABLE adds a new column or row

5. The INSERT INTO just selects the columns you want to go into regardless of the table

6. The VALUES are the new rows being added to the columns

7. The sub query selects what you want to come first

8. The ASC puts them in alphabetical order

9. The PRIMARY KEY is a column within a table

10.The FOREIGN KEY is a column within a table and outside the table

11.The REFERENCE gets the column with the same name and combines them together

CREATE TABLE students(

StudentID int NOT NULL,

StudentName varchar(40),

StudentEmail varchar(40),

PersonID int NOT NULL,

PRIMARY KEY(StudentID),

FOREIGN KEY(PersonID)

REFERENCES Persons(PersonID));

INSERT INTO students

VALUES(2, nick2, “[nick2@gmail.com](mailto:nick2@gmail.com)”,2);

SELECT \* FROM Students INNER JOIN Persons ON Persons.PersonID = Student.PersonID

1. The SELECT query selects what you want to come up in the Result grid

2. The CREATE TABLE creates a new tables

3. The varchar limits the amount of rows

4. The ALTER TABLE adds a new column or row

5. The INSERT INTO just selects the columns you want to go into regardless of the table

6. The VALUES are the new rows being added to the columns

7. The sub query selects what you want to come first

8. The ASC puts them in alphabetical order

9. The PRIMARY KEY is a column within a table

10.The FOREIGN KEY is a column within a table and outside the table

11.The REFERENCE gets the column with the same name and combines them together

12. The INNER JOIN combines 2 tables having the column or row