DB Module

NGEE ANN SCHOOL OF INFOCOMM TECHNOLOGY

Other DMLs And DDLs

Databases

Day 3

Prepared by Staff of School of ICT

Topics

- Other DML Statements
  - INSERT
  - UPDATE
  - DELETE
- DDL Statements
  - CREATE TABLE
  - DROP TABLE

Other DMLs

SQL DMLs

- Data Retrieval
  - SELECT (Day 1 – Day 2)
- Data Manipulation
  - INSERT
  - UPDATE
  - DELETE

Inserting a Row Using **INSERT...VALUES**

- **Format:**

  INSERT [INTO] table_name [(column_list)]

  VALUES

  ({ DEFAULT | NULL | expression } [,...n])

- **Adds one row of data at a time**
- **Multiple rows of data will require multiple INSERT statements**

Inserting a Row Using **INSERT...VALUES**

Example 2.49: Insert a new row into the table Book for all columns.

```
INSERT INTO Book (ISBN, Title, YearPublish,
    PublisherID, BookCat)
VALUES ('0330246631', 'Vet in harness', 1975, 6, 'NF')
```

OR:

```
INSERT INTO Book
VALUES ('0330246631', 'Vet in harness', 1975, 6, 'NF')
```

Inserting a Row Using **INSERT...VALUES** with **NULL**

Example 2.50: Insert a new row into the Book table supplying data for all the mandatory columns: ISBN and Title.

```
INSERT INTO Book (Title, ISBN)
VALUES ('Inside SQLServer 2000', '0735609985')
OR:
INSERT INTO Book
VALUES (‘0735609985’, ‘Inside SQLServer 2000’, NULL,
    NULL, NULL)
```

Inserting Rows Using **INSERT...SELECT**

Example 2.51:

Insert into a separate table, FictionBook, some of the data from all the rows in table Book whose BookCat is 'Fiction'.

Assume that FictionBook has the following columns - ISBN, Title, YearPublish, PublisherID.

Inserting Rows Using **INSERT...SELECT**

```
INSERT INTO FictionBook
        SELECT ISBN, Title, YearPublish, PublisherID
        FROM Book
        WHERE BookCat = 'F'
INSERT INTO FictionBook (Title, YearPublish, ISBN,
    PublisherID)
        SELECT Title, YearPublish, ISBN, PublisherID
        FROM Book
        WHERE BookCat = 'F'
```

Using UPDATE Statement

- **Format:**

  UPDATE { table_name }
  SET { column_name = { expression | DEFAULT | NULL }
  { [FROM { < table_source > } [,...n]]
  [WHERE < search_condition >] }

- **Changes data in existing rows**
- **Updates single or multiple columns**
- **Updates single or multiple rows**
- ***When WHERE clause is omitted, all rows will be updated***

Updating All Rows

Example 2.53: Increase the rental rates of all copies of the books by 10 percent.

```
UPDATE BookCopy
SET RentalRate = RentalRate * 1.1
```

Updating Specific Rows

Example 2.54: Change the address of the Rose Central branch to 33, Rose Central.

```
UPDATE Branch
SET Address = '33, Rose Central'
WHERE Address LIKE '%Rose Central%'
```

Updating Multiple Columns

Example 2.55: Change the address of the Tulip branch to '535 NP Orchard Road' and its telephone number to 64601111.

```
UPDATE Branch
SET Address = '535 NP Orchard Road',
TelNo = '64601111'
WHERE Address LIKE '%Tulip%'
```

Using DELETE Statement

- **Format**

  DELETE table_name

  [FROM table_sources]

  [WHERE search_condition]

- ***Removes one or more rows from a table***
- ***When WHERE clause is omitted, all rows will be removed***

Deleting Specific Rows

Example 2.56: Delete the rows associated with loans made by the member named 'Tan Mei Ling' and delete the details of this member as well.

```
DELETE Loan
FROM Member
WHERE Loan.MemberID = Member.MemberID AND
        Name = 'Tan Mei Ling'
DELETE Member
WHERE Name = 'Tan Mei Ling'
```

Deleting All Rows

Example 2.57: Delete all the rows in the table Loan.

```
DELETE Loan
```

Subqueries in UPDATE, DELETE, and INSERT Statements

Example 2.56:

```
DELETE Loan
FROM Member
WHERE Loan.MemberID = Member.MemberID AND
        Name = 'Tan Mei Ling'
```

Example 2.56a:

```
DELETE Loan
WHERE MemberID =
    (SELECT MemberID
     FROM Member
     WHERE Name = 'Tan Mei Ling')
```

SQL - Data Definition Language

Recap – Intro to SQL

- Structured Query Language
  - A *data access language*, not a programming language
- Consists of 2 main parts
  - *Data Manipulation Language* (*DML*)
    - To retrieve and update data
      - SELECT, INSERT, UPDATE, DELETE
  - *Data Definition Language* (*DDL*)
    - To define data structure and control access to database
      - CREATE TABLE, CREATE VIEW, grant access rights

Topics

- SQL Identifiers
- Data Types
- Data Integrity Controls
- DDL Statements
  - CREATE TABLE
  - DROP TABLE

Rules for naming SQL Identifiers

- Must start with an alphabet.
- Can contain alphabets, numerals, special characters (@, $, # and _).
- Cannot contain spaces.
- May be up to 30 characters in length.
- Must not be a SQL reserved word.

Commonly used Data Types

- char(n), varchar(n)
- int, smallint, tinyint, decimal(p,s)
- money, smallmoney
- datetime, smalldatetime

Data Integrity Controls

- Required Data Integrity
- Domain Integrity
- Entity Integrity
- Referential Integrity
- Enterprise or User-Defined Integrity

Required Data Integrity

Example 2.60: To specify that the column Name of the Member table cannot have NULL values, we define the column as:

Name varchar(50) NOT NULL

Domain Integrity

Example 2.61: To specify that the domain of the column Gender of the Member table is a single character string consisting of either 'M' or 'F', we define the column as:

Gender char(1) NOT NULL CHECK (Gender IN ('M','F'))

Entity Integrity

Example 2.62: Define MemberID as the primary key of the table Member:

PRIMARY KEY (MemberID)

Example 2.63: Define the composite key (ISBN, CopyNo) as the primary key of the table BookCopy:

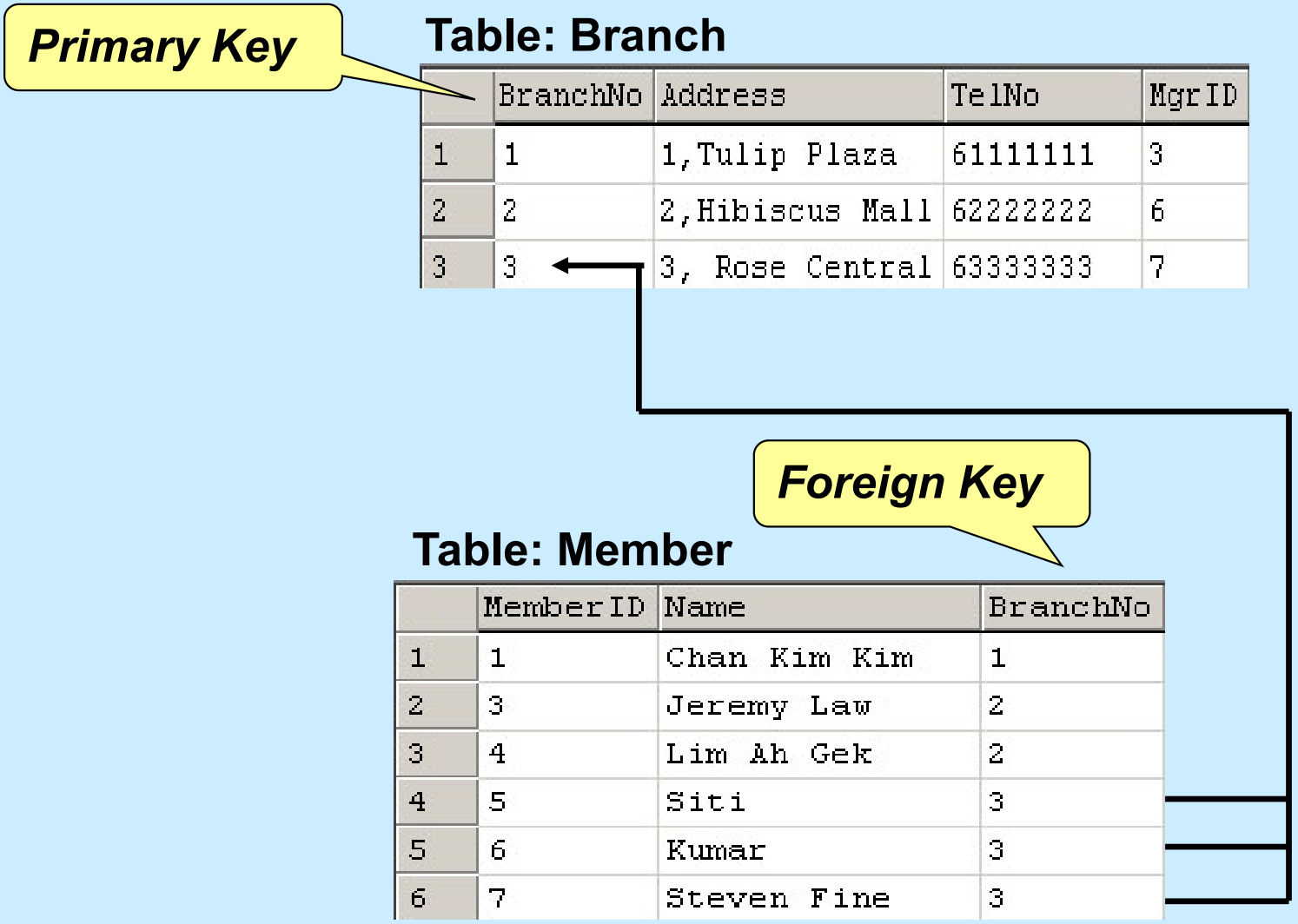
PRIMARY KEY (ISBN, CopyNo)

Unique Alternate Keys

Example 2.64: Define EmailAddr as a unique alternate key for the Member table:

```
EmailAddr varchar(50) NULL UNIQUE
```

Referential Integrity



	BranchNo	Address	TelNo	MgrID
1	1	1,Tulip Plaza	61111111	3
2	2	2,Hibiscus Mall	62222222	6
3	3	3, Rose Central	63333333	7

	MemberID	Name	BranchNo
1	1	Chan Kim Kim	1
2	3	Jeremy Law	2
3	4	Lim Ah Gek	2
4	5	Siti	3
5	6	Kumar	3
6	7	Steven Fine	3

Figure 2.53 Relationship between Branch & Member

Referential Integrity

- Example 2.65: Define BranchNo of table Member as a foreign key:

FOREIGN KEY (BranchNo) REFERENCES Branch(BranchNo)

Tables

Names of Tables

Name	Valid? If No, why?
Emp03	Yes
03Emp	No, starts with a numeral
Fixed_Assets	Yes
Fixed Assets	No, contains space
Order	No, SQL Reserved Word

CREATE TABLE

- **Format:**

  CREATE TABLE table_name

  (column_name format [column_constraint]

  {,column_name format [column_constraint]}

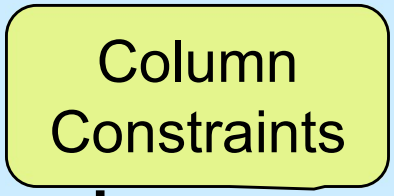
  [,table_constraint {,table_constraint}]

)

CREATE TABLE – Simplest

Example 2.66:

```
CREATE TABLE FictionBook
(
  ISBN            char (10)        PRIMARY KEY,
  Title           varchar (200)    NOT NULL,
  YearPublish     int              NULL,
  PublisherID     smallint         NULL
    FOREIGN KEY REFERENCES Publisher(PublisherID)
 )
```



CREATE TABLE – More Complex

Example 2.67:

```
CREATE TABLE Member
(
  MemberID      int,
  Name          varchar (50)      NOT NULL,
  Address       varchar (150)     NULL,
  ContactNo     char (10)         NULL,
  EmailAddr     varchar (50)      NULL UNIQUE,
  Gender        char (1)          NOT NULL CHECK (Gender IN ('M', 'F')),
  DateJoin      datetime          NOT NULL DEFAULT (GETDATE()),
  BranchNo      tinyint           NOT NULL,
  CONSTRAINT PK_Member PRIMARY KEY (MemberID),
  CONSTRAINT FK_Member_BranchNo
     FOREIGN KEY (BranchNo) REFERENCES Branch(BranchNo)
)
```

Table Constraints

Table Constraint

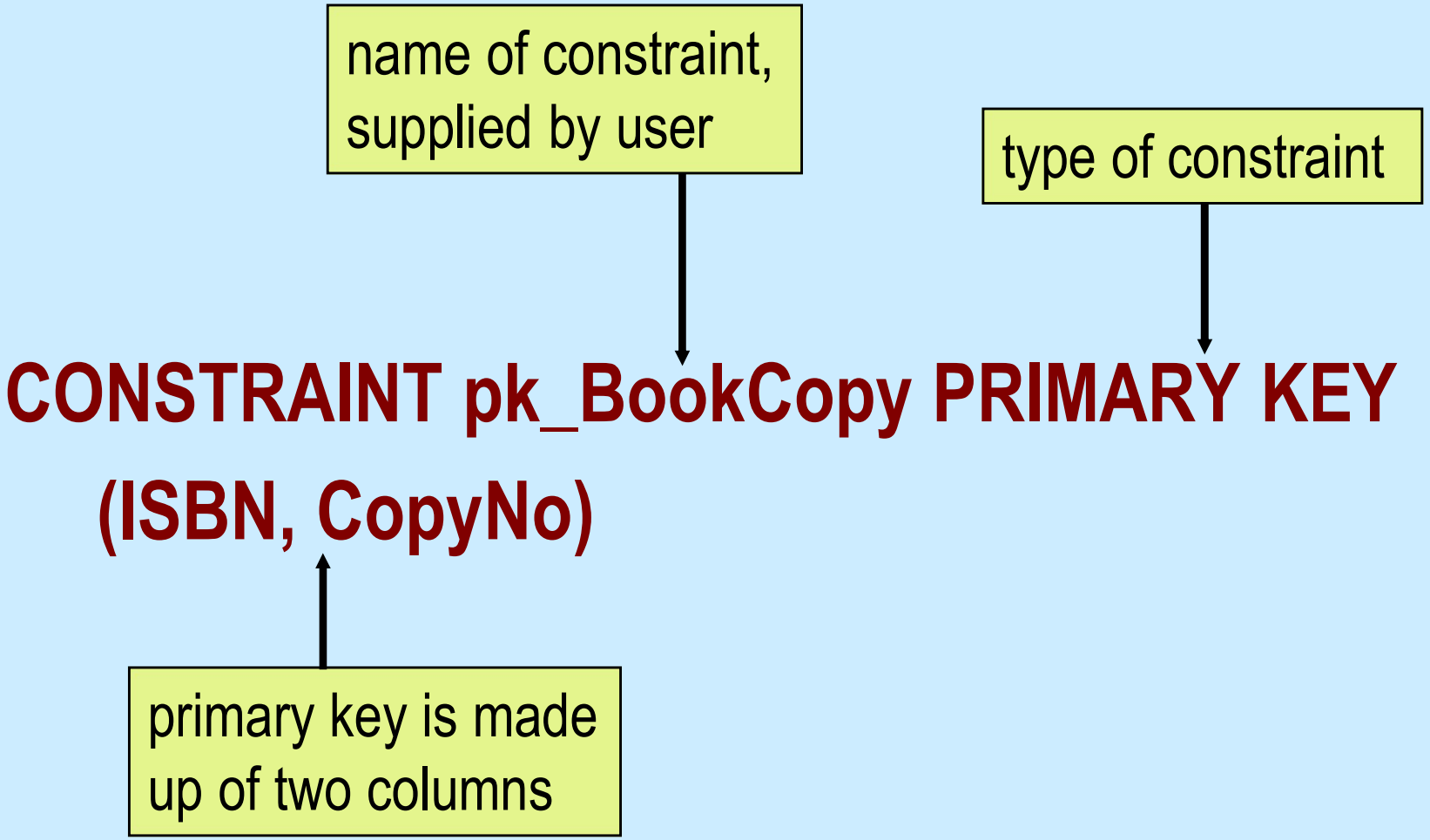
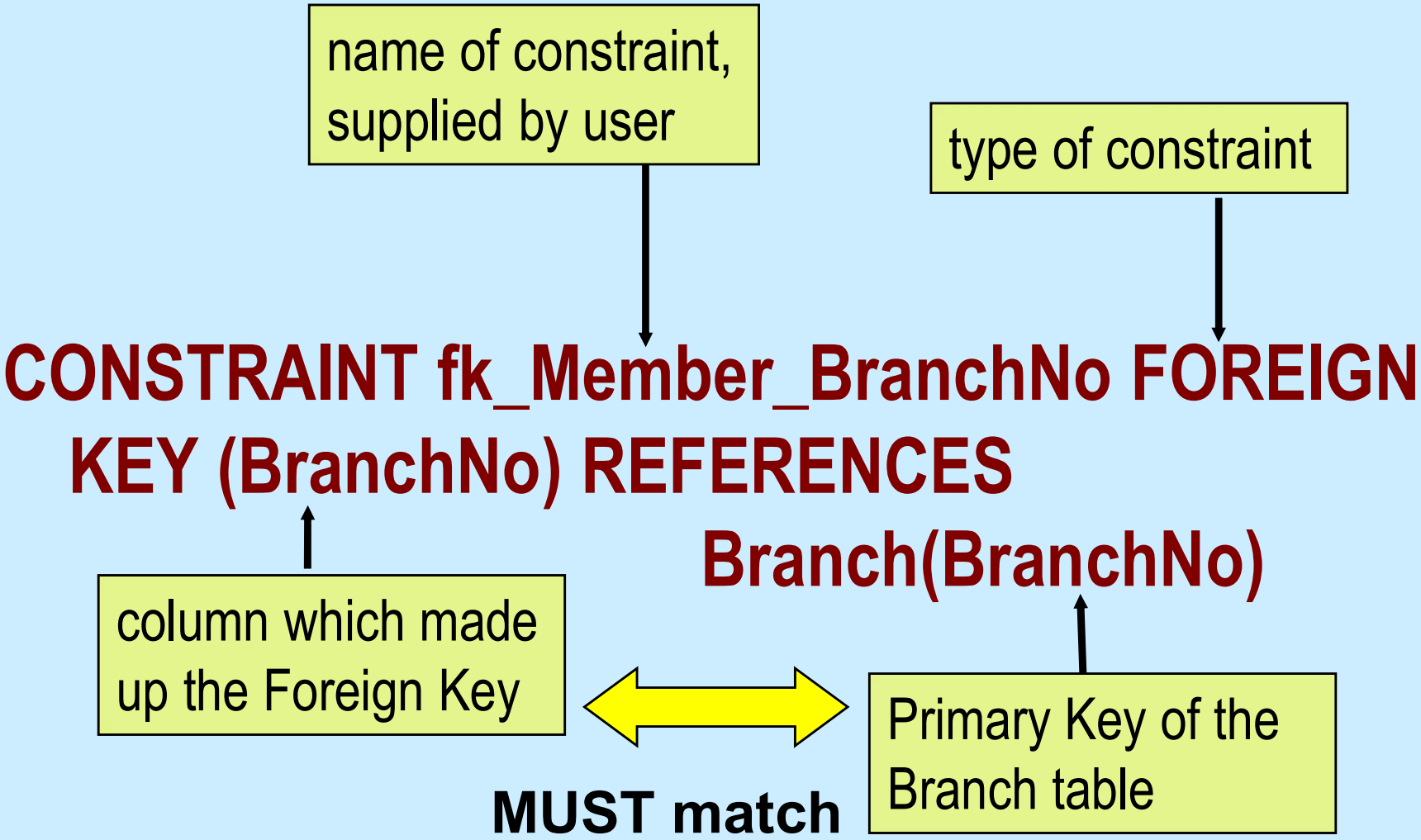


Table Constraint



Destroy Existing Table

- Format:

  DROP TABLE table_name
- must be table owner or DBA
- Example 2.72: Remove the table FictionBook.

  DROP TABLE FictionBook

Compare with

  DELETE FictionBook

any Difference?

Concat function

- CONCAT takes a variable number of string arguments and concatenates them into a single string. It requires a minimum of two input values.
- http://msdn.microsoft.com/en-us/library/hh231515.aspx
- Format:

  CONCAT (string_value1, string_value2 [, string_valueN])

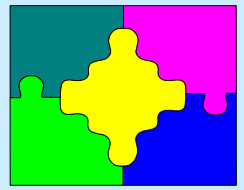
Concat function

- **Example:**

```
SELECT CONCAT(sup.name, ' is the supevisor of ',
s.name)
FROM staff s INNER JOIN staff sup
ON s.SupervisorID = sup.StaffID
```

	(No column name)
1	Mary is the supevisor of Richard
2	Richard is the supevisor of John
3	Jane is the supevisor of Sun Sun
4	Nana is the supevisor of Jane
5	May May is the supevisor of Sadiah
6	May May is the supevisor of Samuel

Summary



- Other DML Statements
  - INSERT
  - UPDATE
  - DELETE
- SQL Identifiers
- Data Types
- Data Integrity Controls
- DDL Statements
  - CREATE TABLE
  - DROP TABLE