

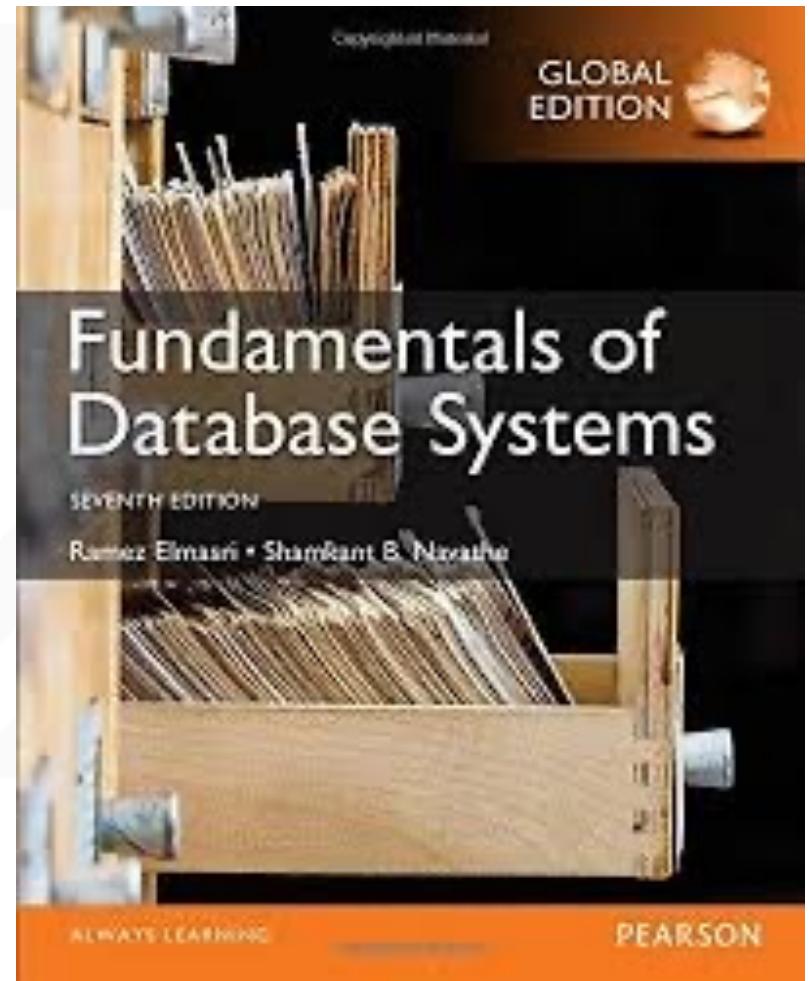
# Database Systems

Program in Computer Engineering  
Faculty of Engineering

King Mongkut's Institute of Technology Ladkrabang

# Text

- Ramez Elmasri and Shamkant B. Navathe.  
“Fundamentals of Database Systems”  
7<sup>th</sup> Edition., Pearson, 2017



# Chapter 6

## Basic SQL

# Outline

- SQL Data Definition and Data Types
- Specifying Constraints in SQL
- Basic Retrieval Queries in SQL
- **INSERT, DELETE, and UPDATE Statements in SQL**
- Additional Features of SQL

# Basic SQL

## • SQL language

- Considered one of the major reasons for the commercial success of relational databases
- Now popularly known as “**Structured Query language**”.
- SQL is an **informal or practical rendering** of the relational data model with syntax

Quriesy and insert, update, delete

# SQL Data Definition, Data Types, Standards

- Terminology:
  - **Table**, **row**, and **column** used for relational model terms  
**relation**, **tuple**, and **attribute**  
<sup>Domain</sup>
- **CREATE statement**
  - Main SQL command for data definition
  - The language has features for :  
Data definition, Data Manipulation, Transaction control (Transact-SQL, Ch. 20),  
Indexing (Ch.17), Security specification (Grant and Revoke- see Ch.30),  
Active databases (Ch.26), Multi-media (Ch.26), Distributed databases (Ch.23) etc.

# SQL Standards

- SQL has gone through many standards: starting with SQL-86 or SQL 1.A. SQL-92 is referred to as SQL-2.
- Later standards (from **SQL-1999**) are divided into **core** specification and specialized **extensions**. The extensions are implemented for different applications – such as data mining, data warehousing, multimedia etc.  
*and data type ฯ ลักษณะ*
- SQL-2006 added **XML** features (Ch. 13); In 2008 they added Object-oriented features (Ch. 12).
- **SQL-3** is the current standard which started with SQL-1999. It is not fully implemented in any RDBMS.

กําหนด  
มาตรฐาน

SQL 1999

# Schema and Catalog Concepts in SQL

- **SQL schema**

1 DBMS වල Database යොමුවක

- Identified by a **schema name**
- Includes an **authorization identifier** and **descriptors** for each element

- **Schema elements** include

- Tables, constraints, views, domains, and other constructs
- **Each statement in SQL ends with a semicolon (;)**

ជំនួយការងារក្នុង CREATE

- **CREATE SCHEMA statement**

- CREATE SCHEMA COMPANY  
AUTHORIZATION 'Jsmith'; ដែលគឺជាឯុទ្ធសាស្ត្រ Jsmith

- **Catalog** ការណែនាំ TABLE នៃប្រព័ន្ធ

- Named collection of schemas in an SQL environment
- SQL also has the concept of a cluster of catalogs.

# The CREATE TABLE Command in SQL

- Specifying a new relation
  - Provide name of table
  - Specify attributes, their types and initial constraints

- Can optionally specify schema:

ดีที่สุด • CREATE TABLE COMPANY.<sup>ระบุชื่อ schema</sup>EMPLOYEE ...<sup>TABLE</sup>

or

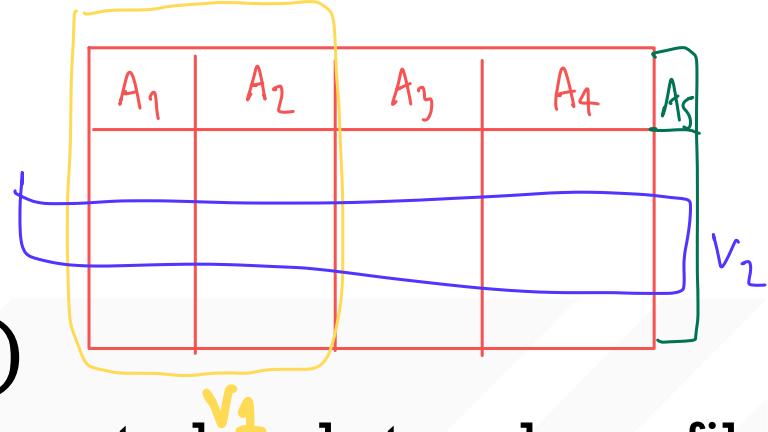
• CREATE TABLE EMPLOYEE ...

เลือกไปแล้วก็เขียนต่อได้

DBMS จะ TABLE เป็นอย่างไร Database ชื่อว่า Company  
กัน / ป้องกันการผิดพลาด

$V_1$  access 2 table និង 1 base table

$V_2$  access 1 table + ផ្លូវ



- Base table ឯក
- រាយការណ៍
- រាយការណ៍

## • Base tables (base relations)

- Relation and its tuples are actually created and stored as a file by the DBMS

## • Virtual relations (views)

- Created through the CREATE VIEW statement. Versioned version នឹងការប្រើប្រាស់  
Do not correspond to any physical file.

ស្នើសុំ **relation** នៃ table នេះ

អាមេរិកសាពី user ទៅវិវឌ្ឍន៍ / លើកការក្នុង

បញ្ជាផ្ទាល់ condition នៃការណា ដូចជាបញ្ជាក់ពីរប់ប៉ុណ្ណោះ

នៃ TA នៃលក្ខណៈ នាយកឱ្យ view ត្រូវការក្រោមពារ update នូវលក្ខណៈ / និង AT នៃវិថី

max cardinality

ក្នុងការប្រើប្រាស់ user នាយកឱ្យ នឹង SW

វិធី

នៅពេលការប្រើប្រាស់ SW នៅក្នុងការប្រើប្រាស់  
អាមេរិកសាពី SW នៃក្នុងការប្រើប្រាស់

11

# COMPANY relational database schema (Fig. 5.7)

**EMPLOYEE**

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

**DEPARTMENT**

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

**DEPT\_LOCATIONS**

<u>Dnumber</u>	Dlocation
----------------	-----------

**PROJECT**

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

**WORKS\_ON**

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

**DEPENDENT**

<u>Essn</u>	Dependent_name	Sex	Bdate	Relationship
-------------	----------------	-----	-------	--------------

# One possible database state for the COMPANY relational database schema (Fig. 5.6)

**EMPLOYEE**

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

**DEPARTMENT**

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

**DEPT\_LOCATIONS**

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

### **WORKS\_ON**

<u>Essn</u>	<u>Pno</u>	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

### **PROJECT**

<u>Pname</u>	<u>Pnumber</u>	<u>Plocation</u>	<u>Dnum</u>
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

### **DEPENDENT**

<u>Essn</u>	<u>Dependent_name</u>	<u>Sex</u>	<u>Bdate</u>	<u>Relationship</u>
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

# SQL CREATE TABLE data definition statements for defining the COMPANY schema from Figure 5.7 (Fig. 6.1)

สร้าง table		
CREATE TABLE EMPLOYEE		
( Fname	VARCHAR(15) <span style="color: red;">ก. ภาษาไทยต้อง</span>	Set Not null ไม่ต้อง ต้อง
Minit	CHAR,	NOT NULL,
Lname	VARCHAR(15)	NOT NULL,
Ssn	CHAR(9)	NOT NULL,
Bdate	DATE,	
Address	VARCHAR(30),	
Sex	CHAR,	
Salary	DECIMAL(10,2),	
Super_ssn	CHAR(9),	
Dno	INT	
Set PRIMARY KEY (Ssn),	NOT NULL,	
CREATE TABLE DEPARTMENT		
( Dname	VARCHAR(15)	1 NOT NULL,
Dnumber	INT	NOT NULL,
Mgr_ssn	CHAR(9)	NOT NULL,
Mgr_start_date	DATE,	
PRIMARY KEY (Dnumber),		
UNIQUE (Dname), <span style="color: red;">2 ห้องนี้จะ</span>		
FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn);		
CREATE TABLE DEPT_LOCATIONS		
( Dnumber	INT	NOT NULL,
Dlocation	VARCHAR(15)	NOT NULL,
PRIMARY KEY (Dnumber, Dlocation),		
FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber);		

ສິນສໍາກົດລົດຕ້ອງ

Foreign key ລັກນີ້ສະໝັບ table ທີ່ refer  
ໄດ້ error ຖໍ່ມີການ table ທີ່ພືບເປີ

Set

PRIMARY KEY (Ssn),

CREATE TABLE DEPARTMENT

( Dname  
    Dnumber  
    Mgr\_ssn  
    Mgr\_start\_date

ການຫຼັບມົນ  
ມັນສີ, ໃຊ້ຈຳຕັ້ງ ຖໍ່ໃຫຍໍ ອຳນວຍກົດລົດຕ້ອງ  
ໃນ table ບໍ່ມີ, ຢຸ່ງທຸນ

1 NOT NULL,  
NOT NULL,  
NOT NULL,

table — table  
ກົດລົດຕ້ອງສຳພັນນຳກົນ

**CREATE TABLE PROJECT**

( Pname	VARCHAR(15)	<b>NOT NULL,</b>
Pnumber	INT	<b>NOT NULL,</b>
Plocation	VARCHAR(15),	
Dnum	INT	<b>NOT NULL,</b>

**PRIMARY KEY** (Pnumber),  
**UNIQUE** (Pname),  
**FOREIGN KEY** (Dnum) **REFERENCES** DEPARTMENT(Dnumber) );

**CREATE TABLE WORKS\_ON**

( Essn	CHAR(9)	<b>NOT NULL,</b>
Pno	INT	<b>NOT NULL,</b>
Hours	DECIMAL(3,1)	<b>NOT NULL,</b>

**PRIMARY KEY** (Essn, Pno),  
**FOREIGN KEY** (Essn) **REFERENCES** EMPLOYEE(Ssn),  
**FOREIGN KEY** (Pno) **REFERENCES** PROJECT(Pnumber) );

**CREATE TABLE DEPENDENT**

( Essn	CHAR(9)	<b>NOT NULL,</b>
Dependent_name	VARCHAR(15)	<b>NOT NULL,</b>
Sex	CHAR,	
Bdate	DATE,	
Relationship	VARCHAR(8),	

**PRIMARY KEY** (Essn, Dependent\_name),  
**FOREIGN KEY** (Essn) **REFERENCES** EMPLOYEE(Ssn) );

## • Some foreign keys may cause errors

- Specified either via:
  - Circular references ✓ 
  - Or because they refer to a table that has not yet been created 
- DBA's have ways to stop referential integrity enforcement to get around this problem.

ອໝາວືນ foreign key ວິຊາ

ກົດຕົກ TABLE ຫຼັກສິນ

ເລີຍທຳໃນສ່ວນຂອງ foreign keys

# Attribute Data Types and Domains in SQL

- Basic **data types**

- **Numeric** data types

- **Integer numbers:** INTEGER, INT, and SMALLINT
  - **Floating-point (real) numbers:** FLOAT or REAL, and DOUBLE PRECISION

(0-๙๙, 0-๖๖ — ॥ສ້າງຕົກລວງໆ)

- **Character-string** data types

- **Fixed length:** CHAR (n), CHARACTER (n)
  - **Varying length:** VARCHAR (n), CHAR VARYING (n), CHARACTER VARYING (n)

ໃນ VARCHAR[15]

ນາສູນສູນໄຟໂທ 15 ສຳເນົາ

- **Bit-string** data types
  - **Fixed length:** BIT ( $n$ )
  - **Varying length:** BIT VARYING ( $n$ )
- **Boolean** data type
  - Values of TRUE or FALSE or NULL
- **DATE** data type
  - Ten positions *(สิบจุดที่ต้องระบุตั้งแต่ปีเดือนวัน)*
  - Components are YEAR, MONTH, and DAY in the form YYYY-MM-DD
  - Multiple mapping functions available in RDBMSs to change date formats

## • Additional data types

### • **Timestamp** data type

Includes the DATE and TIME fields

- Plus a minimum of six positions for decimal fractions of seconds

- Optional WITH TIME ZONE qualifier

HH : MM : SS . dd mm yy

กราบว่า รูปนี้ ก็จะดู เด่นๆ

### • **INTERVAL** data type

timestamp  
มีนาคม

- Specifies a relative value that can be used to increment or decrement an absolute value of a date, time, or timestamp

set เวลา ถ้าต้องการ / 1 นาที

- DATE, TIME, Timestamp, INTERVAL** data types can be **cast** or converted to string formats for comparison.

๑๕๘๙๒ - ๖๗๘๙๙๖๓

## • **Domain**

datatype ຈຸດໜາກີ່ທີ່ນີ້ ສໍາກຳ domain ຖັນ

- Name used with the attribute specification
- Makes it easier to change the data type for a domain that is used by numerous attributes
- Improves schema readability
- Example:
  - CREATE DOMAIN SSN\_TYPE AS CHAR(9);  
char 9 ມີ

## • **TYPE**

- User Defined Types (UDTs) are supported for object-oriented applications. (See Ch.12) Uses the command: CREATE TYPE

ສໍາເລັດ

# Specifying Constraints in SQL

## Basic constraints:

- Relational Model has 3 basic constraint types that are supported in SQL:
  - **Key constraint**: A primary key value cannot be duplicated
  - **Entity Integrity Constraint**: A primary key value cannot be null ก็จะเป็น null
  - **Referential integrity** constraints : The “foreign key “ must have a value that is already present as a primary key, or may be null.

ពីរនេះនឹង ត្រូវដោយ

# Specifying Attribute Constraints

## Other Restrictions on attribute domains:

- Default value of an attribute
  - **DEFAULT** <value>
  - **NULL** is not permitted for a particular attribute (**NOT NULL**)
- **CHECK clause** *នរោតសម្រាប់ការពិនិត្យចំណាំសំនួរបោះពីរ*
  - Dnumber INT NOT NULL CHECK  
(Dnumber > 0 AND Dnumber < 21);       $1 \leq Dnumber \leq 20$   
 $(1\dots 20)$

# Specifying Key and Referential Integrity Constraints

- **PRIMARY KEY clause**

- Specifies one or more attributes that make up the primary key of a relation
- Dnumber INT PRIMARY KEY;

- **UNIQUE clause**

- Specifies alternate (secondary) keys (called **CANDIDATE** keys in the relational model).
- Dname VARCHAR (15) UNIQUE; *มีค่าเดียว*

ถ้า Attribute มีค่าเดียวในหนึ่ง

ค่าเดียว → super key

ค่าเดียว → minimal key, key ธรรมดาก

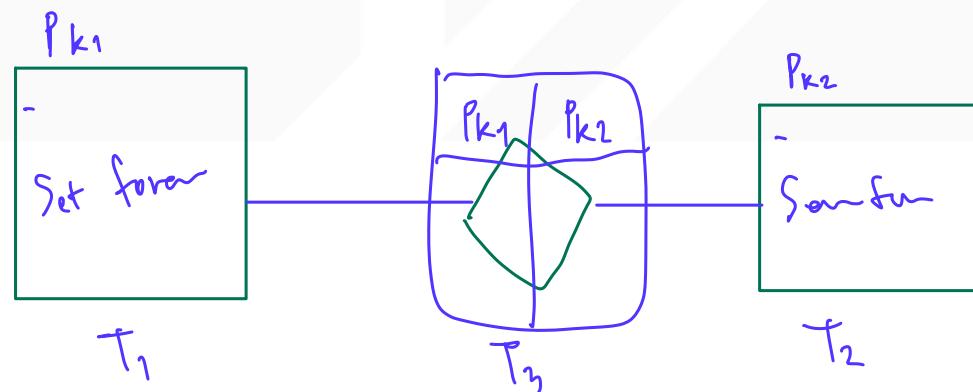
Set key นี้叫做 “Minimal Key”

“Candidate”

## • FOREIGN KEY clause

ກົດພິບໃນຈີ່ໃຊ້ເລີກຂໍ້ວັດວິທີ

- Default operation: **reject update on violation**
- Attach **referential triggered action** clause  
ມານີ້ແກ້ໄຂ
  - Options include SET NULL, CASCADE, and SET DEFAULT
  - Action taken by the DBMS for SET NULL or SET DEFAULT is the same for both ON DELETE and ON UPDATE
  - **CASCADE option suitable for “relationship” relations**



# Giving Names to Constraints

- Using the keyword **CONSTRAINT**
  - Name a constraint
  - Useful for later altering

# Default attribute values and referential integrity triggered action specification (Fig. 6.2)

```

CREATE TABLE EMPLOYEE
(
    ...,
    Dno      INT      NOT NULL      DEFAULT 1,
    CONSTRAINT EMPPK
        PRIMARY KEY (Ssn),
    CONSTRAINT EMPSUPERFK
        FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
            ON DELETE SET NULL      ON UPDATE CASCADE,
    CONSTRAINT EMPDEPTFK
        FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
            ON DELETE SET DEFAULT  ON UPDATE CASCADE);
CREATE TABLE DEPARTMENT
(
    ...,
    Mgr_ssn CHAR(9)      NOT NULL      DEFAULT '888665555',
    ...,
    CONSTRAINT DEPTPK
        PRIMARY KEY(Dnumber),
    CONSTRAINT DEPTSK
        UNIQUE (Dname),
    CONSTRAINT DEPTMGRFK
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
            ON DELETE SET DEFAULT  ON UPDATE CASCADE);
CREATE TABLE DEPT_LOCATIONS
(
    ...,
    PRIMARY KEY (Dnumber, Dlocation),
    FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
        ON DELETE CASCADE      ON UPDATE CASCADE);

```

# Specifying Constraints on Tuples Using CHECK

- Additional Constraints on individual tuples within a relation are also possible using CHECK
- CHECK clauses at the end of a CREATE TABLE statement
  - Apply to each tuple individually
  - **CHECK (Dept\_create\_date <= Mgr\_start\_date);** *check នៅលើការណា*

ឯកតា គម្រោង នៅក្នុង របៀប និង អនុវត្តន៍

# Basic Retrieval Queries in SQL

ຫັນ ສົບ ຂັງ ດອລົມ

- **SELECT statement**

- One basic statement for retrieving information from a database
- SQL allows a table to have two or more tuples that are identical in all their attribute values
  - Unlike relational model (relational model is strictly set-theory based)
  - Multiset or bag behavior
  - Tuple-id may be used as a key

ກິນ້າ ມາລາ ອານ ອັກ

ຍັງວົວມະນຸມວັນນີ້

# The SELECT-FROM-WHERE Structure of Basic SQL Queries

- Basic form of the **SELECT** statement:

**SELECT**      <attribute list>  
**FROM**        <table list>  
**WHERE**      <condition>; ມາດຕະຖານ, department ໄທນີ້ໃສ, ພິຈານີ່ນີ້, ອັນດີ່ງນີ້  
where                  ຖໍ່ມີວິທີ where ສຳຄັນ ທົ່ວນວດ

- <attribute list> is a list of attribute names whose values are to be retrieved by the query.
- <table list> is a list of the relation names required to process the query.
- <condition> is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query.

- **Logical comparison operators**

- $=$ ,  $<$ ,  $\leq$ ,  $>$ ,  $\geq$ , and  $\neq$  ไม่เท่ากัน

- **Projection attributes** ลักษณะที่จะเลือก

- Attributes whose values are to be retrieved

- **Selection condition** บรรทัดที่ จะเลือก

- Boolean condition that must be true for any retrieved tuple.  
Selection conditions include join conditions (see Ch.8) when multiple relations are involved.

# Basic Retrieval Queries

**Query 0.** Retrieve the birth date and address of the employee(s) whose name is 'John B. Smith'.

```
Q0:   SELECT      Bdate, Address
      FROM        EMPLOYEE
      WHERE       Fname='John' AND Minit='B' AND Lname='Smith';
```



Bdate	Address
1965-01-09	731 Fondren, Houston, TX

**Query 1.** Retrieve the name and address of all employees who work for the 'Research' department.

```
Q1:   SELECT      Fname, Lname, Address
      FROM        EMPLOYEE, DEPARTMENT
      WHERE       Dname='Research' AND Dnumber=Dno;
```



Fname	Lname	Address
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX
Joyce	English	5631 Rice, Houston, TX

↑  
↑  
எனதோடு வீர் தலை

விடுமலாண்மேயிலோகாப்பீக் FK கூடுமிக்குள்

**Query 2.** For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birth date.

Q2:    **SELECT**    Pnumber, Dnum, Lname, Address, Bdate  
         **FROM**    PROJECT, DEPARTMENT, EMPLOYEE  
         **WHERE**    Dnum=Dnumber **AND** Mgr\_ssn=Ssn **AND**  
                 Plocation='Stafford';



(c)

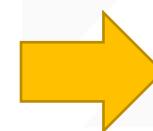
Pnumber	Dnum	Lname	Address	Bdate
10	4	Wallace	291Berry, Bellaire, TX	1941-06-20
30	4	Wallace	291Berry, Bellaire, TX	1941-06-20

# Ambiguous Attribute Names

- Same name can be used for two (or more) attributes in different relations
  - As long as the attributes are in different relations
  - Must **qualify** the attribute name with the relation name to prevent ambiguity

Q1A: **SELECT** Fname, EMPLOYEE.Name, Address  
**FROM** EMPLOYEE, DEPARTMENT  
**WHERE** DEPARTMENT.Name='Research' AND  
DEPARTMENT.Dnumber=EMPLOYEE.Dnumber;

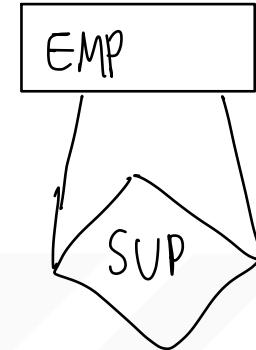

 table      Dnumber      Attribute



Fname	Lname	Address
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX
Joyce	English	5631 Rice, Houston, TX

# Aliasing, and Renaming

Select FName AS First Name



- **Aliases or tuple variables**

- Declare alternative relation names E and S to refer to the EMPLOYEE relation twice in a query:

↑  
Tip

**Query 8.** For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor.

↑  
Q&A

**SELECT** E.Fname, E.Lname, S.Fname, S.Lname  
**FROM** EMPLOYEE AS E, EMPLOYEE AS S *using Python*  
**WHERE** E.Super\_ssn=S.Ssn



E.Fname	E.Lname	S.Fname	S.Lname
John	Smith	Franklin	Wong
Franklin	Wong	James	Borg
Alicia	Zelaya	Jennifer	Wallace
Jennifer	Wallace	James	Borg
Ramesh	Narayan	Franklin	Wong
Joyce	English	Franklin	Wong
Ahmad	Jabbar	Jennifer	Wallace

**Note** Recommended practice to abbreviate names and to prefix same or similar attribute from multiple tables.

- The attribute names can also be renamed

**FROM** EMPLOYEE AS E(Fn, Mi, Ln, Ssn, Bd,  
Addr, Sex, Sal, Sssn, Dno)

จัดชื่อ Attribute

- Note that the relation EMPLOYEE now has a variable name E which corresponds to a tuple variable
- The “AS” may be dropped in most SQL implementations

จัดชื่อ วน AS ที่จะ

# Unspecified WHERE Clause and Use of the Asterisk

- Missing WHERE clause

- Indicates no condition on tuple selection

Q9:    **SELECT**      Ssn  
**FROM**         EMPLOYEE;

$$A = \{a_1, a_2\}$$

$$B = \{b_1, b_2, b_3\}$$

$$A \times B = \{ (a_1, b_1), (a_1, b_2), (a_1, b_3), (a_2, b_1), (a_2, b_2), (a_2, b_3) \}$$

የሽን where የEmployee ነንግብ ሲሆን

- If more than one relation is specified in the FROM clause and there is no WHERE clause, effect is a CROSS PRODUCT (x)
- Result is all possible tuple combinations

Q10:    **SELECT**      Ssn, Dname  
**FROM**         EMPLOYEE, DEPARTMENT;

የሽን በattribute ነው → \*

- Specify an asterisk (\*)

- Retrieve all the attribute values of the selected tuples
- The \* can be prefixed by the relation name; e.g., EMPLOYEE \*

Q1C: **SELECT** \*  
**FROM** EMPLOYEE  
**WHERE** Dno=5;

Q1D: **SELECT** \*  
**FROM** EMPLOYEE, DEPARTMENT  
**WHERE** Dname='Research' **AND** Dno=Dnumber;

Q10A: **SELECT** \*  
**FROM** EMPLOYEE, DEPARTMENT;

*Attribute για τις*

# Tables as Sets in SQL

- SQL does not automatically eliminate duplicate tuples in query results
  - For aggregate operations (See sec 7.1.7) duplicates must be accounted for
  - Use the keyword **DISTINCT** in the SELECT clause
    - Only distinct tuples should remain in the result

Ting

**Query 11.** Retrieve the salary of every employee (Q11) and all distinct salary values (Q11A).    

Q11: **SELECT ALL Salary FROM EMPLOYEE;** *សរុបតម្លៃ* *បញ្ជាក់សរុបតម្លៃ*

Q11A: **SELECT** **DISTINCT** Salary → សរុបលទ្ធផលនេះ មានតម្លៃ 10000 និង 20000  
**FROM** EMPLOYEE;

## • Set operations

- $\cup$  UNION,  $A - B$  EXCEPT (difference), INTERSECT  $\cap$

- Corresponding multiset operations:

UNION ALL, EXCEPT ALL, INTERSECT ALL)

- Type compatibility is needed for these operations to be valid

Tanw

**Query 4.** Make a list of all project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as a manager of the department that controls the project.

**Q4A:** (SELECT DISTINCT Pnumber  
 FROM PROJECT, DEPARTMENT, EMPLOYEE  
 WHERE Dnum=Dnumber AND Mgr\_ssn=Ssn  
 AND Lname='Smith')

Manager

UNION  
 (SELECT DISTINCT Pnumber  
 FROM PROJECT, WORKS\_ON, EMPLOYEE  
 WHERE Pnumber=Pno AND Essn=Ssn  
 AND Lname='Smith');

Employee

# Substring Pattern Matching and Arithmetic Operators

คุณรู้ว่าเป็นต้องเนื่องจากสิ่งที่ก็พอ

- **LIKE** comparison operator

SQL

- Used for string **pattern matching**
- % replaces an arbitrary number of zero or more characters
- underscore (\_) replaces a single character
- Examples:

**WHERE** Address **LIKE** '%Houston,TX%';

**WHERE** Ssn **LIKE** '\_ \_ 1 \_ 8901';

Address ระบุ ที่พำน  
ที่ตั้ง ถูกต้อง  
\_\_\_\_\_ ลงท้าย  
หลักที่ 1

\* arbitrary 0..n -  
\* .jpg ไม่

- **BETWEEN** comparison operator

- E.g., in Q14 :

**WHERE (Salary BETWEEN 30000 AND 40000) AND Dno = 5**

(Salary > 30000 and Salary < 40000)

# Arithmetic Operations

- Standard arithmetic operators:
  - Addition (+), subtraction (-), multiplication (\*), and division (/) may be included as a part of **SELECT**

- **Query 13.** Show the resulting salaries if every employee working on the **③ 'ProductX' project** is given a 10 percent raise.  
**①**      **④**

```
SELECT E.Fname, E.Lname, 1.1 * E.Salary AS Increased_sal
FROM EMPLOYEE AS E, WORKS_ON AS W, PROJECT AS P
WHERE E.Ssn=W.Essn AND W.Pno=P.Pnumber AND P.Pname='ProductX';
```

Social Security number in Link

# Ordering of Query Results

- Use **ORDER BY** clause

- Keyword **DESC** ລາຍລຳອັນດີ ມາດຕະປະລົງ to see result in a descending order of values
- Keyword **ASC** ຜິດພາບ ມີຄວາມປົກການ to specify ascending order explicitly
- Typically placed at the end of the query

ORDER BY D.Dname DESC, E.Lname ASC, E.Fname ASC

*ກ່ອນຈົງ default ສະ*

# Basic SQL Retrieval Query Block

```
SELECT      <attribute list>
FROM        <table list> TABLE ห้ามชื่อภาษาไทย
[ WHERE     <condition> ]
[ ORDER BY   <attribute list> ]; รูปแบบ → 210 | 211 → 210
```

# INSERT, DELETE, and UPDATE Statements in SQL

- Three commands used to modify the database:

- **INSERT, DELETE, and UPDATE**

ເຊີ້ນ

- **INSERT** typically inserts a tuple (row) in a relation (table)

ກົບກວດ

- **UPDATE** may update a number of tuples (rows) in a relation (table) that satisfy the condition

- **DELETE** may also update a number of tuples (rows) in a relation (table) that satisfy the condition

# INSERT

ທຳນັ້ນສົກລະນະ ອະ Table ຜົງຈະໄດ້

ບບ 1 DBMS ສັນຕາຍ DATABASE  
2 DATABASE ສາມາຄສົນລາຍ TABLE

- In its simplest form, it is used to add one or more tuples to a relation
- Attribute values should be listed in the same order as the attributes were specified in the **CREATE TABLE** command
- Constraints on data types are observed automatically
- Any integrity constraints as a part of the DDL specification are enforced

# The INSERT Command

- Specify the relation name and a list of values for the tuple. All values including nulls are supplied.

U1:    **INSERT INTO** EMPLOYEE ( <sup>1</sup>  
**VALUES** ( ‘Richard’, ‘K’, ‘Marini’, ‘653298653’, ‘1962-12-30’, ‘98  
Oak Forest, Katy, TX’, ‘M’, 37000, ‘653298653’, 4 );

หากสั่งinsert ไม่ได้ int, str จะ reject

♡ หากรูปแบบ ordered ที่ต้องไป Attribute ♡

ໃຫ້ເນື້ອງຈຸດປະກັດທີ່ໄວ້ສົມບັນດາ

- The variation below inserts multiple tuples where a new table is loaded values from the result of a query.

**U3A:** **CREATE TABLE**

```
( Emp_name
    Proj_name
    Hours_per_week )
```

**WORKS\_ON\_INFO**

```
VARCHAR(15),
VARCHAR(15),
DECIMAL(3,1) );
```

**U3B:** **INSERT INTO**

**SELECT**  
**FROM**  
**WHERE**

**WORKS\_ON\_INFO** ( Emp\_name, Proj\_name,  
Hours\_per\_week )  
E.Lname, P.Pname, W.Hours  
PROJECT P, WORKS\_ON W, EMPLOYEE E  
P.Pnumber=W.Pno **AND** W.Essn=E.Ssn;

65069 join

ມີການ Insert

# BULK LOADING OF TABLES

ឧបត្ថរ table ឱ្យឯក

- Another variation of **INSERT** is used for bulk-loading of several tuples into tables
- A new table TNEW can be created with the same attributes as T and using LIKE and DATA in the syntax, it can be loaded with entire data.
- EXAMPLE:

```
CREATE TABLE D5EMPS LIKE EMPLOYEE
( SELECT E.*  
  FROM EMPLOYEE AS E
  WHERE E.Dno=5
  WITH DATA;
```

កែតម្រូវអ្នកតាម EMPLOYEE នៃទូទៅ D5EMPS

ស្តីពី TABLE

) ក្នុងការសែង “Like”  
ដោយបានក្រោម table នេះ

ស្តីពី D5EMPS ដូចនេះតាម EMPLOYEE នៃទូទៅ

D5EMPS

នៅក្នុងវាទាមតួន្យេ employee  
លើវេចនា តិន្នន័យ

# DELETE

ມະນຸຍາ ຍັງ delete ນີ້ condition ຖືກອນ, ຕັ້ງແລ້ວທາຕິດ  
ລົບກຳລົດປາຍ ແກ້ໄຂໂທ Condition ທີ່ກ່ຽວຂ້າງ ໄຊສົ່ງເຄີຍ

- Removes tuples from a relation

- Includes a WHERE-clause to select the tuples to be deleted
- Referential integrity should be enforced
- Tuples are deleted from only **one table** at a time  
(unless **CASCADE** is specified on a referential integrity constraint)
- A missing WHERE-clause specifies that **all tuples** in the relation are to be deleted; the table then becomes an empty table
- The number of tuples deleted depends on the number of tuples in the relation that satisfy the WHERE-clause

# The DELETE Command

- Removes tuples from a relation
  - Includes a WHERE clause to select the tuples to be deleted. The number of tuples deleted will vary.

<b>U4A:</b>	<b>DELETE FROM</b> <b>WHERE</b>	EMPLOYEE TABLE Lname='Brown';
<b>U4B:</b>	<b>DELETE FROM</b> <b>WHERE</b>	EMPLOYEE Ssn='123456789';
<b>U4C:</b>	<b>DELETE FROM</b> <b>WHERE</b>	EMPLOYEE Dno=5;
<b>U4D:</b>	<b>DELETE FROM</b>	EMPLOYEE; <i>remove all 2 rows Employee</i>

# UPDATE

modify tuple

- Used to modify attribute values of one or more selected tuples
- A WHERE-clause selects the tuples to be modified
- An additional SET-clause specifies the attributes to be modified and their new values
- Each command modifies tuples *in the same relation*
- Referential integrity specified as part of DDL specification is enforced

- **Example:**

Change the location and controlling department number of project number 10 to 'Bellaire' and 5, respectively

```
U5:      UPDATE PROJECT
          SET PLOCATION = 'Bellaire', DNUM = 5
          WHERE PNUMBER=10
```

## • Example:

ພົມສະລັບ ໃນ ອົດກຳທີ່ ອານຸມັດວຽກ ທີ່

Give all employees in the 'Research' department a 10% raise in salary.

```

U6: UPDATE EMPLOYEE
      SET SALARY = SALARY *1.1
 WHERE DNO IN ( SELECT DNUMBER
                  FROM DEPARTMENT
                 WHERE DNAME='Research' )
          DNAME LIKE "%research%"
```

$\text{SALARY} = \text{SALARY} * 1.1$   
 $\text{DNO} \in \{ \text{DNUMBER} \mid \text{DNAME} = \text{'Research'} \}$

$\text{DNAME} \in \{ \text{DNAME} \mid \text{DNAME} \text{ like } \% \text{research}\% \}$

- In this request, the modified SALARY value depends on the original SALARY value in each tuple
  - The reference to the SALARY attribute on the right of = refers to the old SALARY value before modification
  - The reference to the SALARY attribute on the left of = refers to the new SALARY value after modification

# Additional Features of SQL

- Techniques for specifying complex retrieval queries (see Ch.7)
- Writing programs in various programming languages that include SQL statements: Embedded and dynamic SQL, SQL/CLI (Call Level Interface) and its predecessor ODBC, SQL/PSM (Persistent Stored Module) (See Ch.10)
- Set of commands for specifying physical database design parameters, file structures for relations, and access paths, e.g., CREATE INDEX

- Transaction control commands (Ch.20)
- Specifying the granting and revoking of privileges to users (Ch.30)
- Constructs for creating triggers (Ch.26)
- Enhanced relational systems known as object-relational define relations as classes. Abstract data types (called User Defined Types- UDTs) are supported with CREATE TYPE
- New technologies such as XML (Ch.13) and OLAP (Ch.29) are added to versions of SQL

# Summary

- **SQL**
  - A Comprehensive language for relational database management
  - Data definition, queries, updates, constraint specification, and view definition
- **Covered :**
  - Data definition commands for creating tables
  - Commands for constraint specification
  - Simple retrieval queries
  - Database update commands

