www.itsci.mju.ac.th/sayan

LEC 01: ทบทวนความรู้เรื่องระบบฐานข้อมูลเชิงสัมพันธ์ (RDBMS)

SAYAN UNANKARD

1/2558

ฐานข้อมูลเชิงสัมพันธ์

- ฐานข้อมูลที่มีโครงสร้างข้อมูลในแบบเชิงสัมพันธ์ ได้รับการพัฒนาขึ้นจากแบบจำลองที่กล่าวถึงความสัมพันธ์ระหว่างข้อมูลที่มีชื่อว่า โมเดลเชิงสัมพันธ์ (relational model) ข้อมูลที่จัดเก็บอยู่ในฐานข้อมูลที่มีโครงสร้างข้อมูลในแบบเชิงสัมพันธ์ จะถูกแยกจัดเก็บออกเป็นหน่วยย่อย ๆ ที่เรียกว่าตาราง
- ข้อมูลจะถูกจัดเก็บในรูปแบบของตาราง ซึ่งภายใน ตารางจะแบ่งออกเป็น แถว และ คอลัมน์

คีย์หลัก (PRIMARY KEY) และ คีย์ร่วม (COMPOSITE KEY)

จากรีเลชัน ของนักศึกษา จะแสดงข้อมูลของนักศึกษา 1 คน ต่อ 1 แถว ดังนั้นจะมีกลุ่มของคอลัมน์ที่ไม่ซ้ำกันเลย และสามารถใช้เจาะจงถึงแถวของนักศึกษาคนใดคนหนึ่งได้ เรียกคอลัมน์นี้ว่า คีย์หลัก

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

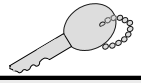
จะเรียกคีย์หลักที่ประกอบด้วยคอลัมน์มากกว่าหนึ่งคอลัมน์ว่า คีย์ร่วม

sid	semester	Gpa	Fee
53666	1/2545	3.4	15,000.00
53666	2/2545	3.2	12,320.00
53650	1/2545	3.8	15,000.00

คีย์คู่แข่ง (CANDIDATE KEYS)

ในรีเลชันทั่วไป อาจจะพบว่ามีคอลัมน์ที่มีคุณสมบัติที่จะเลือกมาเป็นคีย์หลัก จะเรียกกลุ่มคอลัมน์เหล่านี้ว่า คีย์คู่แข่ง

คีย์คู่แข่งที่เหลือหลังจากเลือกคีย์หลักแล้วจะเรียกว่า คีย์สำรอง (Alternate Key)



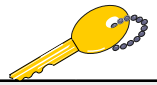
sid	name	idcard	age	gpa
53666	Jones	1523566215223	18	3.4
53688	Smith	2351523624655	18	3.2
53650	Smith	3212025365856	19	3.8

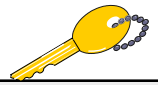
คีย์นอก (FOREIGN KEYS)

เป็นคีย์ที่ใช้แสดงความสัมพันธ์ระหว่างรีเลชัน ซึ่งเป็นคอลัมน์หรือกลุ่มของคอลัมน์ที่อยู่ใน รีเลชันหนึ่งๆ ที่ค่าของคอลัมน์นั้นๆ ไปปรากฏเป็นคีย์หลักในอีกรีเลชันหนึ่ง โดยคีย์นอกและคีย์หลักของอีกรีเลชันหนึ่งต้องอยู่ในโดเมนเดียวกัน

คีย์นอกเป็นคีย์ที่ใช้อ้างถึงข้อมูลในตารางอื่น มักมีชื่อและชนิดข้อมูลเดียวกับคีย์หลักของตารางที่ต้องการอ้างถึง

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8





sid	semester	Gpa	Fee
53666	1/2545	3.4	15,000.00
53666	2/2545	3.2	12,320.00
53650	1/2545	3.8	15,000.00

ENTITY - RELATIONSHIP MODEL

เป็นการนำเสนอโครงสร้างของฐานข้อมูลในระดับความคิด (Conceptual Level) ออกมาในลักษณะของแผนภาพ (Diagram)

เอนติตี้ คือ บางสิ่งที่มีความสำคัญทางธุรกิจที่เกี่ยวกับข้อมูลที่จะต้องรู้ เป็นชื่อที่สามารถลงรายการได้ โดยปกติจะเป็นคำนาม

แอททริบิวส์ คือ ค่าข้อมูลเดี่ยวที่เป็นรายละเอียดของ เอนติตี้ ซึ่งอาจจะเป็นคำอธิบาย จำนวน คุณภาพ กลุ่ม เฉพาะเจาะจง

ความสัมพันธ์ เป็นการอธิบายความสัมพันธ์ระหว่างเอนติตี้ โดยปรกติจะมี 2 มุมมอง จะใช้คำกริยาในการกำหนดความสัมพันธ์

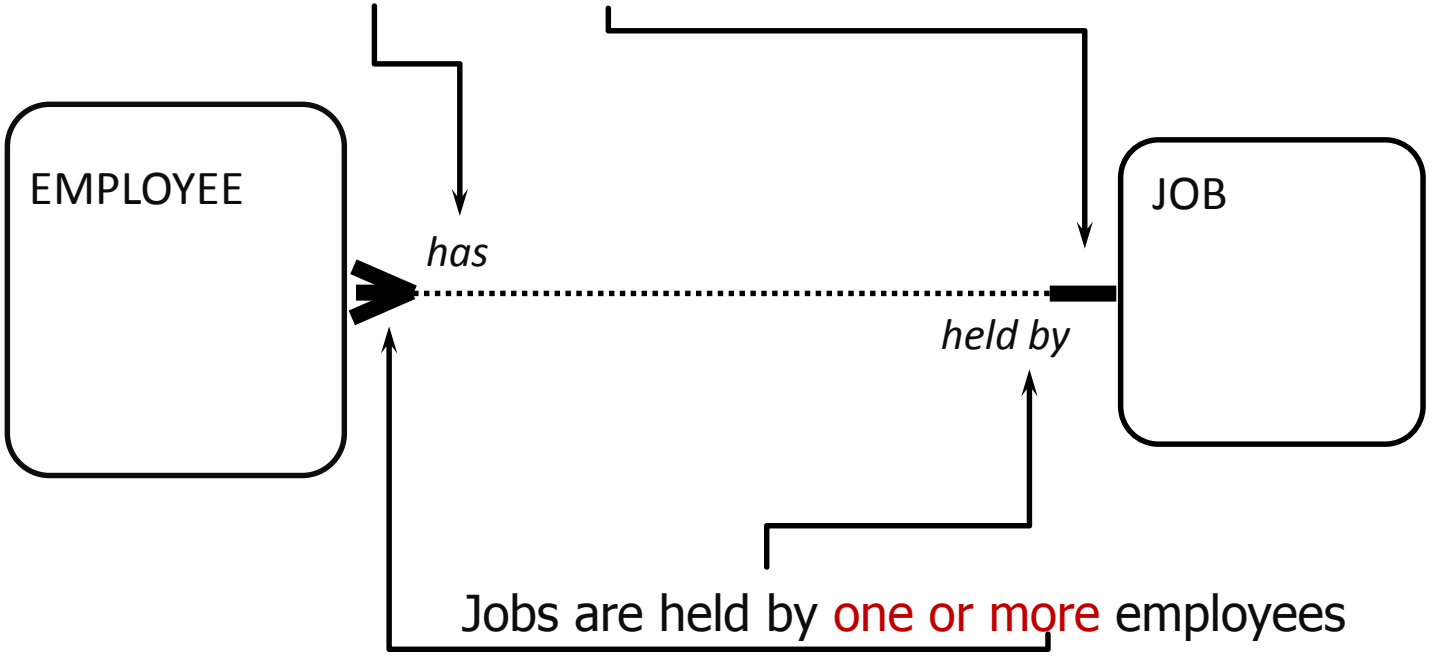
ตัวอย่างของ แอททริบิวส์

Entity	Attribute
EMPLOYEE	Family Name, Age, Shoe Size,
	Town of Residence, Email, ...
CAR	Model, Weight, Catalog Price, ...
ORDER	Order Date, Ship Date, ...
JOB	Title, Description, ...
TRANSACTION	Amount, Transaction Date, ...
EMPLOYMENT CONTRACT	Start Date, Salary, ...

ตัวอย่างการเขียนความสัมพันธ์

An employee has exactly one job.

พนักงานจะต้องมีงานรับผิดชอบ 1 งาน



Jobs are held by one or more employees

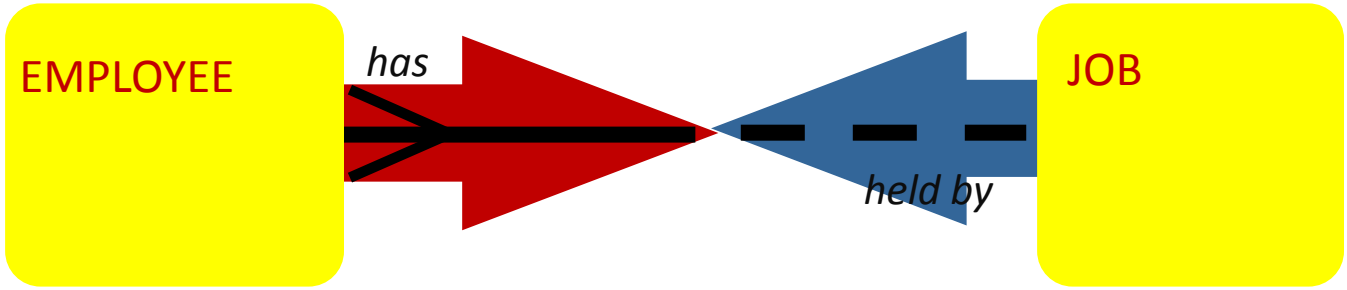
งานจะถูกควบคุมด้วยพนักงานได้หลายคน

การกำหนดความสัมพันธ์

mandatory: ——————— optional: - - - - -

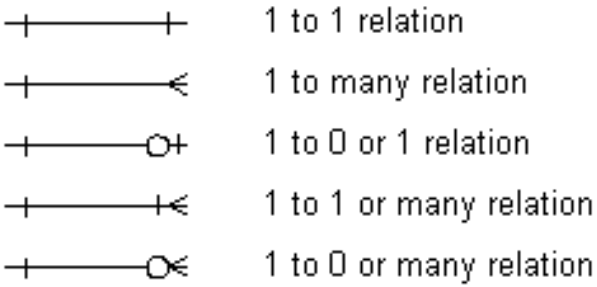
Every EMPLOYEE *has* exactly one JOB

พนักงานทุกคนจะต้องมีงาน 1 งาน



A JOB may be *held by* one or more

งานอาจจะถูกควบคุมโดยพนักงาน 1 หรือ หลายคน



DATABASE OBJECTS

Object	Description
Table	Basic unit of storage; composed of rows
View	Logically represents subsets of data from one or more tables
Sequence	Generates numeric values
Index	Improves the performance of some queries
Synonym	Gives alternative name to an object

NAMING RULES

การตั้งชื่อตารางและคอลัมน์

- จะต้องขึ้นต้นด้วยตัวอักษร
- จะต้องยาวไม่เกิน 30 ตัวอักษร
- ประกอบด้วยอักษร A–Z, a–z, 0–9, _, $, และ # เท่านั้น
- จะต้องไม่ซ้ำกับชื่อของวัตถุอื่นๆ ในผู้ใช้เดียวกัน
- จะต้องไม่ใช่ Oracle server–reserved word

CREATE TABLE STATEMENT

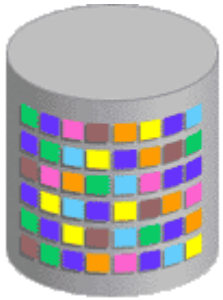
สิ่งที่ต้องมีคือ

- กำหนดสิทธิ์ในการ CREATE TABLE
- พื้นที่ในการจัดเก็บ

```
CREATE TABLE [schema.]table
        (column datatype [DEFAULT expr][, ...]);
```

สิ่งที่ต้องระบุ

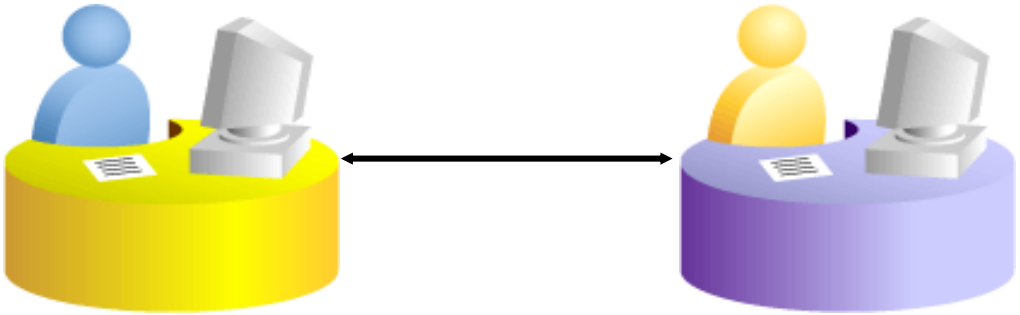
- ชื่อตาราง
- ชื่อคอลัมน์ ชนิดของข้อมูล และ ขนาดของคอลัมน์



REFERENCING ANOTHER USER'S TABLES

ตารางแต่ละตารางจะต้องมีเจ้าของเป็นผู้สร้างตาราง

หากต้องการเรียกใช้ตารางของบุคคลอื่น ๆ จะต้องกำหนดคำนำหน้าชื่อของเจ้าของ ก่อนหน้าตารางเสมอ



USERA

```
SELECT *
FROM userB.employees;
```

USERB

```
SELECT *
FROM userA.employees;
```

DEFAULT OPTION

การกำหนดค่าเริ่มต้นสำหรับคอลัมน์ ระหว่างที่มีการเพิ่มข้อมูลใหม่

```
... hire_date DATE DEFAULT SYSDATE, ...
```

ค่าตามตัวอักษร นิพจน์ หรือ ฟังก์ชัน SQL สามารถกำหนดได้

ชื่อคอลัมน์อื่นๆ หรือ pseudocolumn ไม่สามารถใช้ได้

ชนิดข้อมูลของค่าเริ่มต้นจะต้องตรงกับชนิดข้อมูลของคอลัมน์ นั้น ๆ

```
CREATE TABLE hiredates
        (id          NUMBER(8),
         hire_date DATE DEFAULT SYSDATE);
CREATE TABLE succeeded.
```

```
insert into hiredates(id) values (1);
select * from hiredate;
```

CREATING TABLES

การสร้างตาราง

```
CREATE TABLE dept
        (deptno      NUMBER(2),
         dname       VARCHAR2(14),
         loc         VARCHAR2(13),
         create_date DATE DEFAULT SYSDATE);
```

```
CREATE TABLE succeeded.
```

เรียกดูโครงสร้างของตาราง

```
DESCRIBE dept
```

```
DESCRIBE dept
Name                          Null     Type
----------------------------- -------- ------------
DEPTNO                                 NUMBER(2)
DNAME                                  VARCHAR2(14)
LOC                                    VARCHAR2(13)
CREATE_DATE                            DATE
```

DATA TYPES

Data Type	Description
VARCHAR2(*size*)	Variable-length character data
CHAR(*size*)	Fixed-length character data
NUMBER(*p*,*s*)	Variable-length numeric data
DATE	Date and time values
LONG	Variable-length character data (up to 2 GB)
CLOB	Character data (up to 4 GB)
RAW and LONG RAW	Raw binary data
BLOB	Binary data (up to 4 GB)
BFILE	Binary data stored in an external file (up to 4 GB)
ROWID	A base-64 number system representing the unique address of a row in its table

INCLUDING CONSTRAINTS

การกำหนดเงื่อนไขในระดับตาราง

ช่วยปกป้องจากการลบข้อมูลในตารางหากมีข้อมูลที่ขึ้นอยู่กับข้อมูลอื่น ๆ

ชนิดของเงื่อนไขมีทั้งหมด 5 ชนิด คือ

- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK

http://code.function.in.th/sql/constraint

CONSTRAINT GUIDELINES

การสร้าง Constraint สามารถเพิ่ม ลบ ได้

- ALTER TABLE ชื่อตาราง drop constraint ชื่อเงื่อนไข;
- ALTER TABLE ชื่อตาราง add constraint เงื่อนไข;

สามารถกำหนดชื่อเงื่อนไข หรือ สามารถให้ออราเคิลเซิร์ฟเวอร์ เป็นผู้กำหนดชื่อให้อัตโนมัติ โดยมีรูปแบบคือ SYS_C*n*

สร้างเงื่อนไขเมื่อใด

- สร้างพร้อมกันกับสร้างตาราง
- สร้างหลังจากสร้างตารางแล้ว

กำหนดเงื่อนไขในระดับ คอลัมน์ หรือ ตาราง

สามารถดูเงื่อนไขที่สร้างได้จาก พจนานุกรมข้อมูล (Data Dictionary)

DEFINING CONSTRAINTS

Syntax:

```
CREATE TABLE [schema.]table
      (column datatype [DEFAULT expr]
      [column_constraint],
      ...
      [table_constraint][,...]);
```

Column-level constraint syntax:

```
column [CONSTRAINT constraint_name] constraint_type,
```

Table-level constraint syntax:

```
column,...
  [CONSTRAINT constraint_name] constraint_type (column, ...),
```

DEFINING CONSTRAINTS

Example of a column-level constraint:

```
CREATE TABLE employees(
  employee_id  NUMBER(6)
     CONSTRAINT emp_emp_id_pk PRIMARY KEY,
  first_name    VARCHAR2(20),
  ...);
```

1

Example of a table-level constraint:

```
CREATE TABLE employees(
  employee_id  NUMBER(6),
  first_name    VARCHAR2(20),
  ...
  job_id        VARCHAR2(10) NOT NULL,
  CONSTRAINT emp_emp_id_pk PRIMARY KEY (EMPLOYEE_ID));
```

2

NOT NULL CONSTRAINT

เพื่อให้แน่ใจว่าค่าว่าง (null values) จะไม่ถูกบันทึกลงในคอลัมน์

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	HIRE_DATE	JOB_ID	COMMISSION_PCT
100	Steven	King	SKING	17-JUN-87	AD_PRES	(null)
101	Neena	Kochhar	NKOCHHAR	21-SEP-89	AD_VP	(null)
102	Lex	De Haan	LDEHAAN	13-JAN-93	AD_VP	(null)
103	Alexander	Hunold	AHUNOLD	03-JAN-90	IT_PROG	(null)
104	Bruce	Ernst	BERNST	21-MAY-91	IT_PROG	(null)
107	Diana	Lorentz	DLORENTZ	07-FEB-99	IT_PROG	(null)
124	Kevin	Mourgos	KMOURGOS	16-NOV-99	ST_MAN	(null)
141	Trenna	Rajs	TRAJS	17-OCT-95	ST_CLERK	(null)
142	Curtis	Davies	CDAVIES	29-JAN-97	ST_CLERK	(null)
143	Randall	Matos	RMATOS	15-MAR-98	ST_CLERK	(null)
144	Peter	Vargas	PVARGAS	09-JUL-98	ST_CLERK	(null)
149	Eleni	Zlotkey	EZLOTKEY	29-JAN-00	SA_MAN	0.2
... 174	Ellen	Abel	EABEL	11-MAY-96	SA_REP	0.3

`NOT NULL` constraint (Primary Key enforces `NOT NULL` constraint.)

`NOT NULL` constraint

Absence of `NOT NULL` constraint (Any row can contain a null value for this column.)

NOT NULL CONSTRAINT

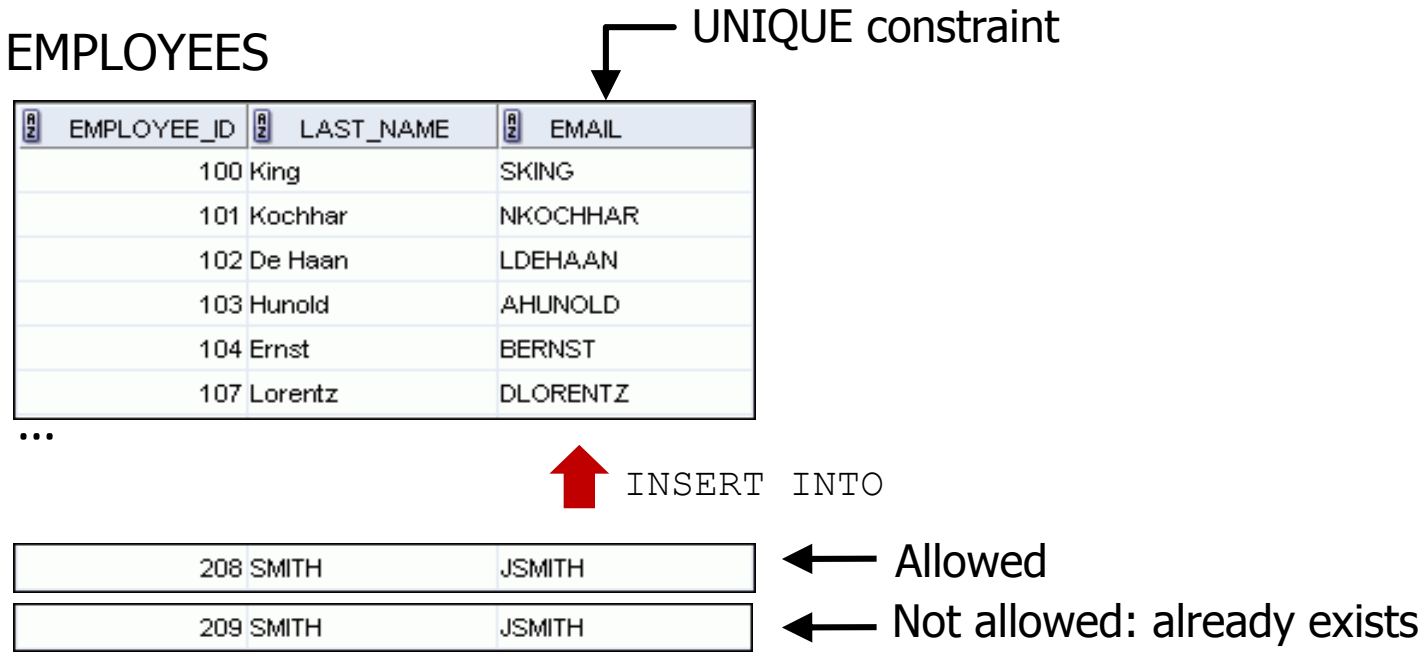
NOT NULL ต้องระบุแบบ column level เท่านั้น

```
CREATE TABLE employees(
    employee_id      NUMBER(6),
    last_name        VARCHAR2(25) NOT NULL,
    email            VARCHAR2(25),
    salary           NUMBER(8,2),
    commission_pct   NUMBER(2,2),
    hire_date        DATE NOT NULL,
...
```

UNIQUE CONSTRAINT

เงื่อนไขการกำหนดไม่ให้บันทึกข้อมูลในคอลัมน์นั้น ๆ ซ้ำ

อนุญาต null



UNIQUE CONSTRAINT

สามารถกำหนดได้ทั้งในระดับ table level หรือ column level

```
CREATE TABLE employees(
    employee_id      NUMBER(6),
    last_name        VARCHAR2(25) NOT NULL,
    email            VARCHAR2(25) ,
    salary           NUMBER(8,2),
    commission_pct   NUMBER(2,2),
    hire_date        DATE NOT NULL,
...
    CONSTRAINT emp_email_uk UNIQUE(email));
```

PRIMARY KEY CONSTRAINT

DEPARTMENTS

PRIMARY KEY

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700

Not allowed (null value)

INSERT INTO

	Public Accounting	124	2500
50	Finance	124	1500

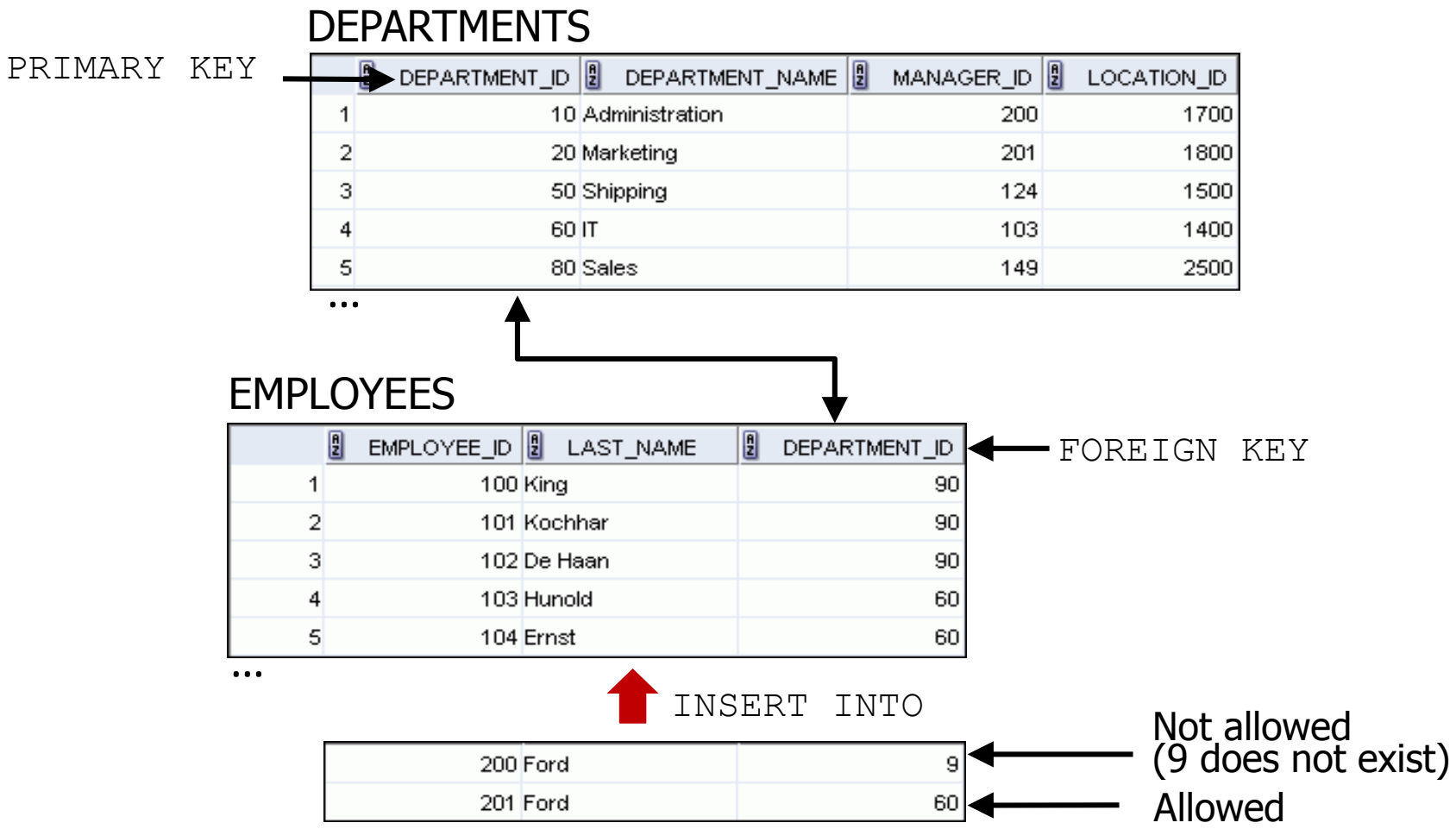
Not allowed (50 already exists)

PRIMARY KEY CONSTRAINT

สามารถกำหนดได้ทั้งในระดับ table level หรือ column level

```
CREATE TABLE employees(
    employee_id      NUMBER(6) ,
    last_name        VARCHAR2(25) NOT NULL,
    email            VARCHAR2(25),
    salary           NUMBER(8,2),
    commission_pct   NUMBER(2,2),
    hire_date        DATE NOT NULL,
...
    department_id    NUMBER(4),
    CONSTRAINT emp_employeeid_pk  PRIMARY KEY (employee_id)
    CONSTRAINT emp_email_uk UNIQUE(email));
```

FOREIGN KEY CONSTRAINT



FOREIGN KEY CONSTRAINT

สามารถกำหนดได้ทั้งในระดับ table level หรือ column level

```
CREATE TABLE employees(
    employee_id      NUMBER(6),
    last_name        VARCHAR2(25) NOT NULL,
    email            VARCHAR2(25),
    salary           NUMBER(8,2),
    commission_pct   NUMBER(2,2),
    hire_date        DATE NOT NULL,
...
    department_id    NUMBER(4),
    CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)
    REFERENCES departments(department_id)  ,
    CONSTRAINT emp_email_uk UNIQUE(email));
```

FOREIGN KEY CONSTRAINT: KEYWORDS

FOREIGN KEY: กำหนดคอลัมน์ในตารางรอง เงื่อนไขในระดับตาราง

REFERENCES: ระบุตารางและคอลัมน์ในตารางหลัก

ON DELETE CASCADE: การลบข้อมูลในตารางหลัก ตารางรองจะถูกลบตามโดยอัตโนมัติ

ON DELETE SET NULL: เปลี่ยนค่า foreign key ในตารางรอง เมื่อตารางหลักถูกลบข้อมูล ให้เป็นค่าว่าง (null)

```
CREATE TABLE employees(
    employee_id      NUMBER(6),
    last_name        VARCHAR2(25) NOT NULL,
...
    department_id    NUMBER(4),
    CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)
    REFERENCES departments(department_id) ON DELETE CASCADE
);
```

CHECK CONSTRAINT

กำหนดเงื่อนไขในแต่ละคอลัมน์ เพื่อให้เข้ากับเงื่อนไขที่กำหนด

นิพจน์ต่อไปนี้ ไม่สามารถใช้ในการกำหนดเงื่อนไขได้

- การอ้างถึงคำสั่ง CURRVAL, NEXTVAL, LEVEL และ ROWNUM
- การเรียกใช้ SYSDATE, UID, USER และ USERENV functions
- การคิวรีข้อมูลที่อ้างไปยังค่าข้อมูลในแถวอื่น ๆ

```
..., salary      NUMBER(2)
    CONSTRAINT emp_salary_min  CHECK (salary > 0),...
```

CREATE TABLE: EXAMPLE

```
CREATE TABLE employees
    ( employee_id    NUMBER(6)
        CONSTRAINT emp_employee_id       PRIMARY KEY
    ,first_name     VARCHAR2(20)
    ,last_name      VARCHAR2(25)
        CONSTRAINT emp_last_name_nn       NOT NULL
    , email          VARCHAR2(25)
        CONSTRAINT     emp_email_nn      NOT NULL
        CONSTRAINT     emp_email_uk      UNIQUE
    , phone_number   VARCHAR2(20)
    , hire_date      DATE
        CONSTRAINT     emp_hire_date_nn  NOT NULL
    , job_id         VARCHAR2(10)
        CONSTRAINT     emp_job_nn        NOT NULL
    , salary         NUMBER(8,2)
        CONSTRAINT     emp_salary_ck     CHECK (salary>0)
    , commission_pct NUMBER(2,2)
    , manager_id     NUMBER(6)
          CONSTRAINT emp_manager_fk      REFERENCES
           employees (employee_id)
    , department_id  NUMBER(4)
        CONSTRAINT     emp_dept_fk       REFERENCES
           departments (department_id));
```

VIOLATING CONSTRAINTS

```
UPDATE employees
SET    department_id = 55
WHERE  department_id = 90;
```

```
Error starting at line 1 in command:
UPDATE employees
SET    department_id = 55
WHERE  department_id = 110
Error report:
SQL Error: ORA-02291: integrity constraint (ORA16.EMP_DEPT_FK) violated - parent key not found
02291. 00000 - "integrity constraint (%s.%s) violated - parent key not found"
*Cause:    A foreign key value has no matching primary key value.
*Action:   Delete the foreign key or add a matching primary key.
```

Department 55 does not exist.

VIOLATING CONSTRAINTS

คำสั่งในการลบข้อมูลนี้ จะไม่สามารถลบข้อมูลได้เนื่องจาก เงื่อนไขของข้อมูลที่เป็นคีย์นอก (department_id) ไปยังตารางอื่น ๆ เนื่องจากไม่ได้กำหนด DELETE CASCADE

```
DELETE FROM departments
WHERE department_id = 60;
```

```
Error starting at line 1 in command:
DELETE FROM departments
WHERE       department_id = 60
Error report:
SQL Error: ORA-02292: integrity constraint (ORA16.EMP_DEPT_FK) violated - child record found
02292. 00000 - "integrity constraint (%s.%s) violated - child record found"
*Cause:    attempted to delete a parent key value that had a foreign
           dependency.
*Action:   delete dependencies first then parent or disable constraint.
```

CREATING A TABLE USING A SUBQUERY

การสร้างตารางโดยเพิ่มข้อมูลไปด้วย ซึ่งจะเป็นการผสมระหว่างคำสั่งในการสร้างตารางและคำสั่งคิวรีข้อมูล ดังรูปแบบ

```
CREATE TABLE table
                [(column, column...)]
AS subquery;
```

จับคู่จำนวนคอลัมน์ของตาราง กับ จำนวนคอลัมน์ของคิวรี

กำหนดชื่อคอลัมน์และค่าเริ่มต้นให้แต่ละคอลัมน์

ปรากฏข้อมูลจากการ select เก็บในตาราง

CREATING A TABLE USING A SUBQUERY

```
CREATE TABLE  dept80
  AS
    SELECT  employee_id, last_name,
            salary*12 ANNSAL,
            hire_date
    FROM    employees
    WHERE   department_id = 80;
```

```
CREATE TABLE succeeded.
```

```
DESCRIBE dept80
```

```
Name                          Null     Type
----------------------------- -------- -------------
EMPLOYEE_ID                            NUMBER(6)
LAST_NAME                     NOT NULL VARCHAR2(25)
ANNSAL                                 NUMBER
HIRE_DATE                     NOT NULL DATE
```

ALTER TABLE STATEMENT

การใช้คำสั่ง ALTER TABLE

- สำหรับเพิ่มคอลัมน์ใหม่ (Add new column)
- สำหรับการแก้ไขคอลัมน์ที่มีอยู่ (Modified column)
- กำหนดค่าเริ่มต้นของคอลัมน์ใหม่
- ลบคอลัมน์ (Drop a column)
- การเปลี่ยนชื่อคอลัมน์ (Rename a column)
- เปลี่ยนตารางให้สามารถอ่านได้อย่างเดียว (read-only)

  - ALTER TABLE .... READ ONLY

RENAMING A TABLE

การเปลี่ยนชื่อตาราง

```
RENAME old_table_name TO new_table_name;
```

การเปลี่ยนชื่อคอลัมน์

```
ALTER TABLE tablename
RENAME COLUMN old_name TO new_name;
```

ALTERING TABLES

การเพิ่มคอลัมน์ในตาราง

```
ALTER TABLE tablename
ADD (fieldname field_specification);
```

Example

```
ALTER TABLE Faculty ADD(start_date Date);
```

ALTERING TABLES

การแก้ไขคอลัมน์

```
ALTER TABLE tablename
MODIFY (fieldname new_field_specification);
```

Example

```
ALTER TABLE Faculty MODIFY(f_name Varchar2(50));
```

ALTERING TABLES

การลบคอลัมน์

```
ALTER TABLE tablename
DROP COLUMN fieldname;
```

Example

```
ALTER TABLE Faculty DROP(start_date);
```

DROPPING A TABLE

คำสั่งในการลบตาราง

ลบทั้งตารางและข้อมูลทั้งหมด ถ้ามีคำสั่ง PURGE clause

ทำให้ objects ที่ขึ้นอยู่กับตารางที่ลบใช้ไม่ได้ และ ถูกลบสิทธิ์ตามไปด้วย

```
DROP TABLE dept80;
```

```
DROP TABLE dept80 succeeded.
```