DATABASE FUNDAMENTALS

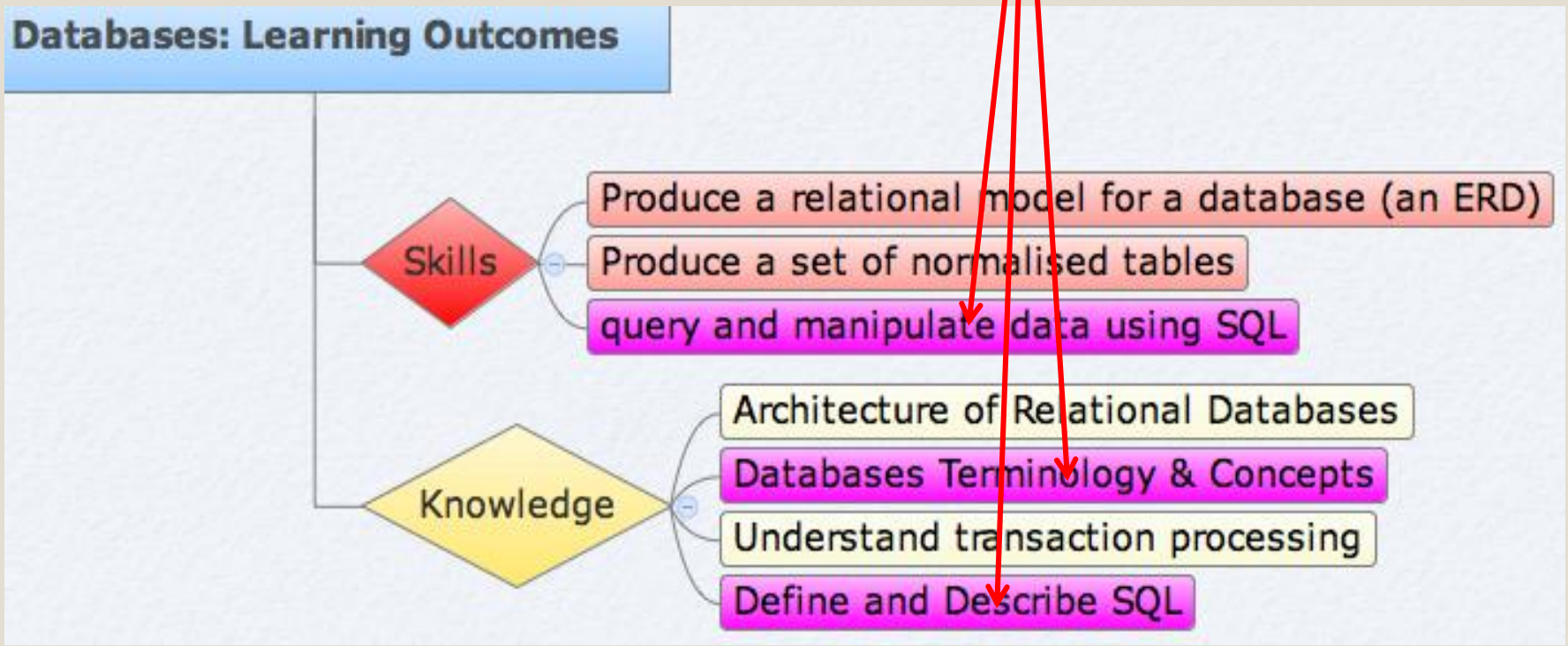
Lecture 5:

Remaining SQL statements:

Create, Alter, Drop, Truncate

Insert, Delete, Update

Learning Outcomes



Objective for this lecture:

Data Definition:

- ‘Creating tables’, Changing the definition of an existing table, deleting tables

Data Manipulation:

- Insert new data, update existing data, deleting rows.

SQL Section 1

CREATING DATABASE OBJECTS

Naming conventions

- Names must begin with a letter
- Names can be 1 - 30 characters long
- Must contain only A-Z, a-z, 0-9, _, $ and #
- Every object owned by a user must have a different name
- Can not use reserved words (as specified by the DBMS implementation)
- Names are NOT case sensitive, so EMP is the same as eMP.

Comments in Code

- There are two ways to specify a comment:

1. -- (two dashes)

1. Multi-line comments

```
/*   This is a comment    */
```

CREATING OBJECTS

- There are a number of different database objects you can create, such as
  - tables
  - views
  - index

Table

RSI Num.	Emp Name	Emp Address	Car Reg	Salary	Grade	Job Title
112541	Banks	Dublin 12	98D1245	20000	4	Engineer
16221	Allen	Dublin 15	99D54421	30000	7	Manager
25541	Hope	Dublin 2	97D844	53000	9	Partner

View

Emp Name	Emp Address	Job Title
Banks	Dublin 12	Engineer
Allen	Dublin 15	Manager
Hope	Dublin 2	Partner

Index

Emp Name	Table Row
Allen	2
Banks	1
Hope	3

- Only the table holds data

Database Objects

Table – Basic **unit of storage** in relational databases

- Holds the **actual raw data** stored in a database

View - **portion** or **subset** of a table provided to a user to perform some task. It's a virtual table.

Index - Like an index in a book. You can build an index over any attribute in the table. The index will contain the list of distinct values for that attribute, and the address of rows in the table that contain that value.

Summary of DDL commands

- CREATE TABLE
  - basic table definition
  - add constraints
- ALTER TABLE
  - add new columns to the table
  - change existing column definitions
  - delete columns from the table
  - change constraints
- DROP TABLE – delete a table
- TRUNCATE TABLE – remove data from a table

Table Creation

- To create a table, use the CREATE command which has the following syntax

```
CREATE TABLE table_name
        (column1 datatype (…),
      (column2 datatype(..),
        ……);
```

- ❖table is the name of the table
- ❖column is the name of the attribute
- ❖datatype is the type and length of the attribute

Remember constraints (lect 2)?

1. Domain constraints
2. Entity integrity constraint
3. NULL constraint
4. Referential integrity constraint

These are defined in the Create statement

Example of CREATE

```
CREATE TABLE dept
        (deptno     INT,
         dname      VARCHAR(14),
         loc        VARCHAR(13)
        );
```

Table name

Attribute domain – data type is character, length is 14.

Attribute name

Example of CREATE

```
CREATE TABLE emp(
 empno INT NOT NULL,
 ename VARCHAR(10),
 job VARCHAR(9),
 mgr  INT,
 hiredate DATE,
 sal DECIMAL(7,2),
 comm DECIMAL(7,2),
 deptno  INT NOT NULL)
```

Table name

Null constraint

Datatypes

Valid Data types for MySQL include:

- VARCHAR(*length)*
- BINARY[(*length)]*
- INTEGER[(*length)]*
- REAL[(*length,decimals)]*
- *DECIMAL[(length[,decimals])]*
- *NUMERIC[(length[,decimals])]]*
- *DATE*
- *TIME*

Full list: http://dev.mysql.com/doc/refman/5.1/en/create-table.html

Exercise

- What is the CREATE statement for the following table:
  - Course (courseID character length 5, course name character length 50, start date of type date)

```
Create table Course
(
CourseID varchar(5),
Coursename char(50),
date     Date,
);
```

Defining the primary key (entity constraint)

Syntax: CONSTRAINT contraint_name PRIMARY KEY (column_name[,column name, . . .])

```
CREATE TABLE DEPT (
 DEPTNO          INT NOT NULL,
 DNAME           VARCHAR(14),
 LOC             VARCHAR(13),
 CONSTRAINT DEPT_PK PRIMARY KEY (DEPTNO));
```

Primary key column must be NOT NULL

The attribute to be used as the primary key

Each constraint is given a name

The type of constraint, i.e. defining a primary key.

Defining a foreign key (referential integrity constraint)

```
CREATE TABLE EMP (
 EMPNO          INT NOT NULL,
 ENAME          VARCHAR(10),
 JOB            VARCHAR(9),
 MGR            INT,
 HIREDATE       DATE,
 SAL            DECIMAL(7,2),
 COMM           DECIMAL(7,2),
 DEPTNO         INT NOT NULL,
 CONSTRAINT EMP_DEPTNO_FK FOREIGN KEY (DEPTNO) REFERENCES DEPT
        (DEPTNO),
CONSTRAINT EMP_EMPNO_PK PRIMARY KEY (EMPNO));
```

Each constraint is given a name

The type of constraint, i.e. defining a foreign key.

The table and attribute the foreign key references

Syntax – foreign key

- The Syntax for defining a foreign key is:

CONSTRAINT contraint_name FOREIGN KEY (column name) REFERENCES table_name (primarykey)

Question Time

Exercise

- Write SQL statements to create the following relations as MySQL tables:

- course (course_id(PK), course_name)

- student(student_ID (PK), student_name, student_address, course_id(FK))

Defining additional domain constraints

- CHECK is used to add additional restrictions to an attribute's domain, for example:

```
CREATE TABLE dept (
  deptno INT CHECK (deptno BETWEEN 10 and 49),
  dname VARCHAR(14),
  loc VARCHAR(13) NOT NULL,
  CONSTRAINT dept_pk PRIMARY KEY (deptno)
);
```

CHECK constraint

- A check constraint can also be defined at the end of the create statements as follows:

```
CREATE TABLE dept (
  deptno INT UNIQUE,
  dname VARCHAR(14),
  loc VARCHAR(13) NOT NULL,
  CONSTRAINT dept_pk PRIMARY KEY (deptno),
CONSTRAINT check_deptno CHECK (deptno BETWEEN 10 and 49)
);
```

More examples of Constraints

```
CREATE TABLE  jobs

( job_id  int AUTO_INCREMENT,

 job_desc varchar(50) NOT NULL DEFAULT 'New Position - title
 not formalized yet',

 min_sal int NOT NULL CHECK (min_sal >= 10000),

 max_sal int NOT NULL CHECK (max_sal <= 25000),

  CONSTRAINT jobs_pk  PRIMARY KEY (job_id)

 );
```

AUTO_INCREMENT– a number that increments automatically everytime a row is added to the table. USE SPARINGLY.

Adding constraints to table columns

- The following are valid constraints in MySQL
  - **NOT NULL** - attribute must have a value
  - **NULL** - attribute allows NULL values
  - **UNIQUE** - column(s) must have a unique value for each row in the table
  - **PRIMARY KEY** - establishes a primary key (PK)
  - **FOREIGN KEY** - establishes a relationship between this column and a column in the referenced table
  - **CHECK** - specify a condition that must be true
  - **DEFAULT** –provide default values for columns
  - **AUTO_INCREMENT** – automatically increments the attribute value for new table rows.

Full Specification: http://dev.mysql.com/doc/refman/5.1/en/create-table.html

ALTERING THE DEFINITION OF AN EXISTING TABLE

Altering the table definition

- Once created, you can add, modify or delete (drop) columns from a table using the ALTER statement

- Statement Syntax

ALTER TABLE table_name
ADD column_name datatype [DEFAULT expr] [,…]);

- Or

ALTER TABLE table_name
MODIFY column_name datatype [DEFAULT expr] [,…]);

- Or

ALTER TABLE table
DROP COLUMN column_name;

Adding a column to a table

- Example:

```
ALTER TABLE dept
ADD  location VARCHAR(20);
```

- Note:
  - The new column will become the last column in the table.
  - If the table has data in it, the new column will default to NULL unless a default value is specified.

Modifying an existing column

- You can change a columns datatype, size and default value.

```
ALTER TABLE dept
MODIFY dname VARCHAR(20) NOT NULL;
```

- Note:
  - A change to the default value only affects new rows to be added
  - Changes the datatype, or reducing the size, should only be done if the column only contains NULL values

Dropping a column

- Example

```
ALTER TABLE dept
DROP COLUMN job;
```

- Note:
  - You can only drop one column at a time
  - The column may contain data when dropped
  - The table must have a least one column left after being altered
  - Once a column is dropped, it can not be recovered

Changing constraints

- The ALTER command can also be used to add, modify, disable, enable or drop constraints.
- Examples:

```
ALTER TABLE dept
  ADD CONSTRAINT dept_loc_fk FOREIGN KEY loc REFERENCES location (loc);

ALTER TABLE emp
  DISABLE CONSTRAINT emp_pk CASCADE;
```

Deleting a table

- The DROP command drops a table as follows:
  - syntax: DROP TABLE table_name
  - Example: DROP TABLE dept;

- Note:
  - This deletes the table and its indexes
  - Views are not dropped, but are now invalid
  - This statement is IRREVERSIBLE

Other commands

- To remove all data from a table:

  TRUNCATE TABLE department;

  The table still exists but all data has been deleted permanently

Exercises

- Modify the department (dept) table to add a check constraint to the deptno ensure it is always less that 100.

ALTER TABLE dept

Add CONSTRAINT check_deptno **CHECK** (deptno <100)

Exercises

- Drop all rows from the department table, but don't delete the table itself.

- Delete the department table.

SQL SECTION 2

Adding, updating and deleting data in tables

Data Manipulation

- INSERT - Add new rows to a table
- UPDATE - Modify existing rows in a table
- DELETE - Remove existing rows from a table

SQL - Add new rows in a table

- To add data to the table, use the INSERT command.
- The syntax is:

```
INSERT INTO table_name [(column_1 [,column_2,…])]
VALUES(value_1 [, value_2 . . . ]);
```

- You need a separate insert statement for every row added
- The order of the values inserted must be in the same order of the columns specified in the column list.
- If you want to insert data in all columns of a table, the column list can be omitted – but values must be ordered in the same way as they appear in the table definition.

INSERT EXAMPLE

```
INSERT INTO dept
VALUES (05, 'development', 'Dublin');
```

Column names not needed, but values must be in correct order

```
INSERT INTO dept (deptno, loc)
VALUES (05, 'Dublin');
```

Column names needed as not all columns were getting a value. dname will be set to NULL

```
INSERT INTO dept (deptno, loc, dname)
VALUES (05, 'Dublin', 'development');
```

Column names needed as values are not ordered according to the table definition.

Note: Non numeric data must be in quotes….most of the time!!

More on Insert

- **AUTO_INCREMENT** column – when a row is inserted in a table with **AUTO_INCREMENT** column, you don't have to specify the **AUTO_INCREMENT** column in the INSERT statement – the value is automatically incremented.

- When a column is omitted in the column list of INSERT, the column is set to NULL or it is set to a default value, if one is defined on the column

- You can use system values such as SYSDATE() or SYSTEM_USER when inserting data

INSERT INTO order (order_ID, date, salesperson)

VALUES ('O_1001', GETDATE(), SYSTEM_USER);

System_user available in SQL Server

Insert -dates

- MySQL recognizes date and time data enclosed in single quotation marks (') in the following format:
  - Date: YYYY-MM-DD or YY-MM-DD. Other delimiters are also accepted, e.g. YY/MM/DD
  - DateTime: YYYY-MM-DD HH:MM:SS or YY-MM-DD HH:MM:SS . As with date above, other delimiters are also accepted.

YY- two digit year – 09, 10

YYYY – four digit year – 2009, 2010

MM – numeric month: 01 to 12

DD – numeric day: 01 to 31

HH: hour in 24 hour clock 00 to 23

MM: minutes – 00 to 59

SS: seconds – 00 to 59

INSERT INTO

- You can populate a table by using a subset of another table with the same structure

```
INSERT INTO emp2
SELECT *
FROM emp
WHERE job = 'clerk';
```

Changing data in a table

- The UPDATE command changes the value of an column:
- The syntax is:

```
UPDATE table_name
SET column_1 = new_value,
 column_2 = new_value, ....
[WHERE condition];
```

Examples of UPDATE

UPDATE emp

SET deptno = 07;

- This statement will set the department number of all employees to 07.

before

emp no	deptno
07	05
13	10
15	05
18	20

after

emp no	dept no
07	07
13	07
15	07
18	07

- To change a specific row, you use the WHERE clause

Examples if UPDATE

UPDATE emp

SET deptno = 20

WHERE empno = 15;

- This statement only changes the deptno for employee 15

before

emp no		deptno
07		05
13		10
15		05
18		20

after

emp no	dept no
07	05
13	10
15	20
18	20

Deleting rows from a table

- The DELETE statement deletes rows permanently from a table.
- The syntax is

  DELETE [FROM] table_name
  [WHERE condition];

- If you don't use a WHERE clause, all rows will be deleted
- You can not delete a row where the primary key is a foreign key in another table,
- Example: You could not delete department 05 if there are employees in the emp table allocated to department 05 **(referential integrity constraint).**

Referential Integrity

Course

Course_id	Title
A123	Maths
B654	Economics
C299	Computing

Student

S_id	Name	Course_id
99111	Tom	A123
99112	Ken	B633
99113	Ray	C299

Referential Integrity Rule broken

B633 does not appear in the Course Tables

Examples of delete

DELETE department;

- All rows in the department table are deleted

DELETE emp

WHERE deptno = 05;

- Deletes all rows where department number is 05, i.e. all employees working in department 05.

before

emp no	deptno
07	05
13	10
15	05
18	20

after

emp no	dept no
13	10
18	20

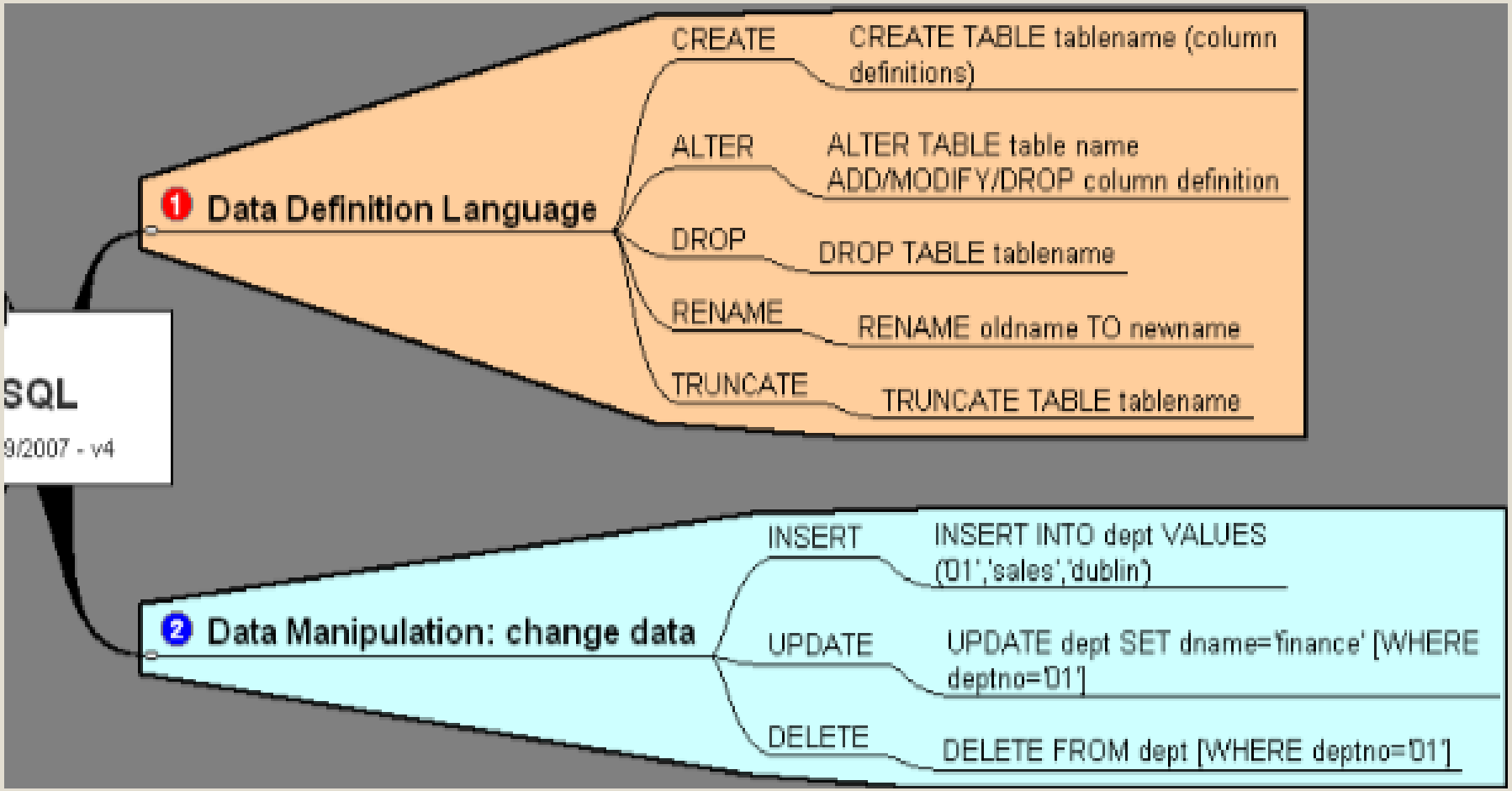
Exercise

- Write SQL statements to do the following
  - **Add a row to the dept table for department 50, sales, based in New York**
  - **Delete the row from the dept table for deptno = 30**
  - **Change department 10 to be based in Washington**

Summary - terminology

- **Database Instance**
  - MySQL implementation include a DBMS and a number of databases
- **Database**
  - A collection of tabes
  - Also called a SCHEMA
- **Table**
  - 2 dimensional structure of rows and column storing information about an entity
  - Also called a RELATION
  - Examples: Studen table; course table;
  - Every table must have a PRIMARY KEY
- **A row**
  - One line in a database table storing information about an entity, e.g. details about ONE student
  - ALso called a TUPLE
  - The CARDINALITY of a table is the number of rows it has
- **Column**
  - All values for a particular attribute, e.g. Age; Name; Date of Birth etc.
  - A DOMAIN defines the data type and valid values allowed for an attribute
  - The DEGREE of a table is the number of columns it has
- **Cell**
  - A single value in a table
- **Primary Key**
  - A columns whos valies unquely identify each row in the table
  - Examples: StudentID; CustomerID; ProductID; email
- **Foreign Key**
  - A column whos value is a Primary Key in another table
  - Foreign keys are used to link tables together
- **Constraints**
  - Integrity constraint: Make sure the primary key is UNIQUE & NOT NULL
  - Referential Integrity: Ensure each foreign key values refers to a valid primary key value in another table
  - Domain contraints: Ensure every value is valid for the domain of that attribute
  - NULL constraint: Don't allow NULLs for attributes that are NOT NULL

Summary - SQL


SQL
9/2007 - v4
1 Data Definition Language
CREATE
CREATE TABLE tablename (column definitions)
ALTER
ALTER TABLE table name ADD/MODIFY/DROP column definition
DROP
DROP TABLE tablename
RENAME
RENAME oldname TO newname
TRUNCATE
TRUNCATE TABLE tablename
2 Data Manipulation: change data
INSERT
INSERT INTO dept VALUES ('01','sales','dublin')
UPDATE
UPDATE dept SET dname='finance' [WHERE deptno='01']
DELETE
DELETE FROM dept [WHERE deptno='01']