



OVERVIEW OF THE SQL QUERY LANGUAGE

• IBM Sequel language developed as part of system R project at the IBM San Jose Research Laboratory
• 1970s
• Renamed Structured Query Language (SQL)
• In 1986, the American National Standards Institute (ANSI) and the International Organization for Standardization (ISO) published an SQL standard, called SQL-86

OVERVIEW OF THE SQL QUERY LANGUAGE

• Structured Query Language (SQL)

• Data-Definition Language, DDL

• Data-Manipulation Language, DML

• Integrity完整性

• View Definition財務定义

• Transaction Control事务控制

• Embedded SQL and dynamic SQL嵌入式和动态SQL

• Authorization授权

• Commercial systems offer most, if not all, SQL-92 features, plus varying feature sets from later standards and special proprietary features.

OBJECTIVES

Overview of The SQL Query Language

Out Definition

Basic Query Structure

Additional Basic Operations

Set Operations

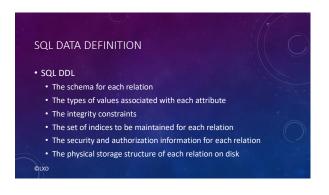
Null Values

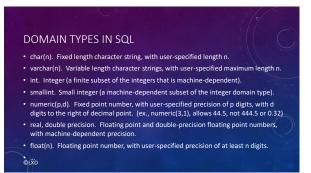
Aggregate Functions

Nested Subqueries

Modification of the Database

OUX





MYSQL DATA TYPES

• https://dev.mysql.com/doc/refman/5.7/en/data-types.html

• Numeric Types

• Date and Time Types

• String Types

• Spatial Data Types

• The JSON Data Type

• Data Type Default Values

BASIC SCHEMA DEFINITION

• Create table
• Insert into
• Delete from
• Alter table
• Drop table

CREATE TABLE CONSTRUCT

• An SQL relation is defined using the create table command: create table r ( $A_1$   $D_4$ ,  $A_2$   $D_2$ , ...,  $A_n$   $D_n$ , (integrity-constraint,完整性约束), ..., (integrity-constraint<sub>k</sub>))

• r is the name of the relation

• each  $A_i$  is an attribute name in the schema of relation r•  $D_i$  is the data type of values in the domain of attribute  $A_i$ 

CREATE TABLE CONSTRUCT

• the greate table command

• Example:

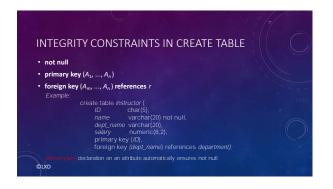
create table instructor (

ID char(5),

name varchar(20),

dept\_name varchar(20),

salary numeric(8,2));

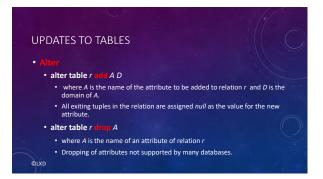




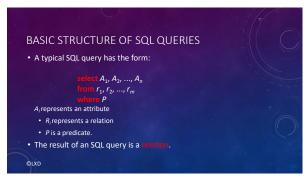












QUERIES ON A SINGLE RELATION

• The select clause lists the attributes desired in the result of a query

• corresponds to the projection operation of the relational algebra

• Example: find the names of all instructors:

select name
from instructor

• NOTE: SQL names are case insensitive
• E.g., Name ≡ NAME ≡ name
• Some people use upper case wherever we use bold font.

QUERIES ON A SINGLE RELATION
- DEDUPLICATION去重
- SQL allows duplicates in relations as well as in query results.
- To force the elimination of duplicates, insert the keyword distinct after select.
- Find the department names of all instructors, and remove duplicates select distinct dept\_name from instructor
- The keyword all specifies that duplicates should not be removed. select all dept\_name from instructor

QUERIES ON A SINGLE RELATION

- MORE

• An asterisk (\*) in the select clause denotes "all attributes" select \* from instructor

• An attribute can be a liveral with no from clause select '437'

• Results is a table with one column and a single row with value "437"

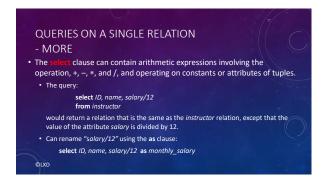
• Can give the column a name using: select '437' as FOO

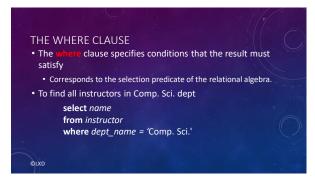
QUERIES ON A SINGLE RELATION

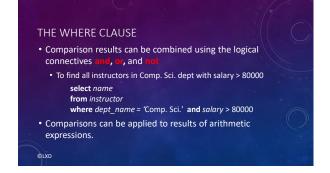
- MORE

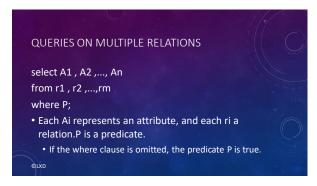
• An attribute can be a literal with from clause select 'A' from instructor

• Result is a table with an explanation and Nows (number of tuples in the instructors table), each row with value "A"

























### THE RENAME OPERATION

- The SQL allows renaming relations and attributes using the **as** clause: old-name **as** new-name
- Find the names of all instructors who have a higher salary than some instructor in 'Comp. Sci'.
  - select distinct T.name from instructor as T, instructor as S where T.salary > S.salary and S.dept\_name = 'Comp. Sci.'
- Keyword **as** is optional and may be omitted instructor **as**  $T \equiv instructor T$

©LXD

## STRING OPERATIONS

- SQL includes a string-matching operator for comparisons on character strings. The operator like uses patterns that are described using two special characters:
  - percent (%). The % character matches any substring.
  - underscore ( \_ ). The \_ character matches any character.

## STRING OPERATIONS

 Find the names of all instructors whose name includes the substring "dar".

> select name from instructor where name like '%dar%'

• Match the string "100%"

like '100 \%' escape '\'

in that above we use backslash (\) as the escape character.

### STRING OPERATIONS

- Patterns are case sensitive.
- Pattern matching examples:
  - 'Intro%' matches any string beginning with "Intro".
  - "%Comp%' matches any string containing "Comp" as a substring.
  - '\_\_\_' matches any string of exactly three characters.
  - '\_\_\_%' matches any string of at least three characters.

©LXI

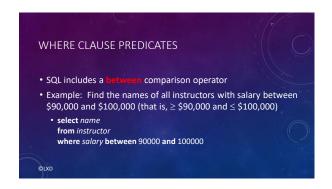
### STRING OPERATIONS

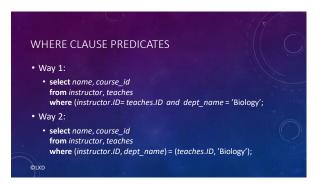
- SQL supports a variety of string operations such as
  - concatenation (using "||")
  - converting from upper to lower case (and vice versa)
  - finding string length, extracting substrings, etc.

©LXD

### ORDERING THE DISPLAY OF TUPLES

- List in alphabetic order the names of all instructors
  - select distinct name from instructor order by name
- We may specify description for descending order or ascending order, for each attribute; ascending order is the default.
  - Example: order by name desc
- Can sort on multiple attributes
- Example: order by dept\_name, name











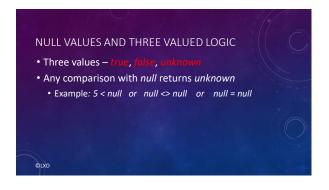


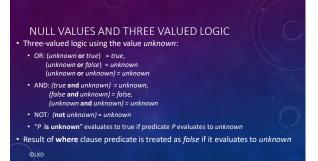




# OBJECTIVES Overview of The SQL Query Language Data Definition Basic Query Structure Additional Basic Operations Set Operations Additional Basic Operations Set Operations Modification of the Database OUND



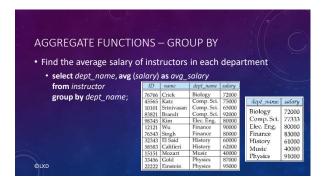


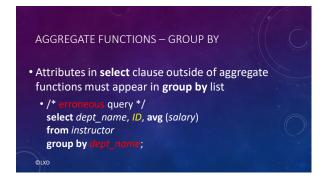












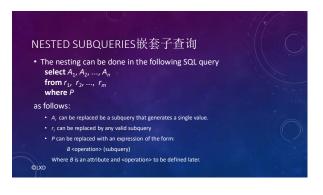


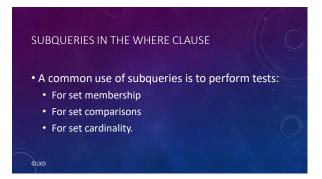




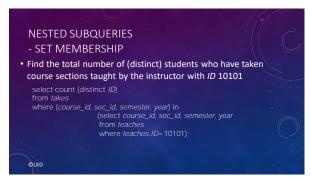


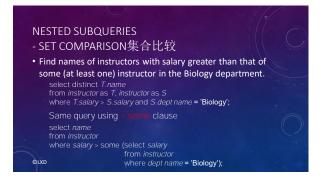




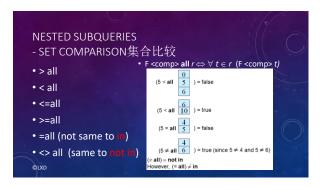




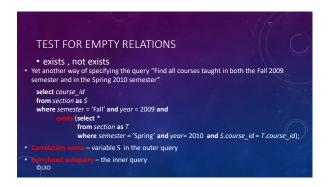


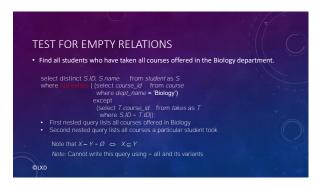










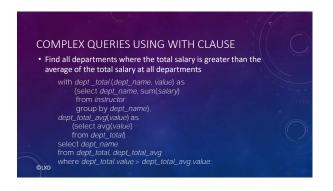


## TEST FOR ABSENCE OF DUPLICATE TUPLES • The unique construct tests whether a subquery has any duplicate tuples in its result. • The unique construct evaluates to "true" if a given subquery contains no duplicates. • Find all courses that were offered at most once in 2009 select T.course\_id from course as T where inque (select R.course\_id from section as R where T.course\_id=R.course\_id and R.year = 2009);



# SUBQUERIES IN THE FROM CLAUSE • Find the average instructors' salaries of those departments where the average salary is greater than \$42,000." • Another way to write above query select dept\_name, avg\_salary from (select dept\_name, avg\_salary) from instructor group by dept\_name) as dept\_avg (dept\_name, avg\_salary) where avg\_salary > 42000;





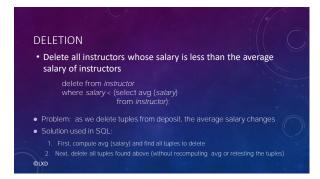




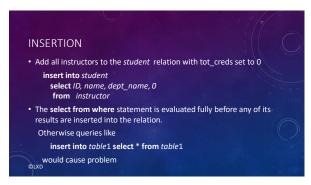
MODIFICATION OF THE DATABASE

• Deletion of tuples from a given relation.
• Insertion of new tuples into a given relation
• Updating of values in some tuples in a given relation

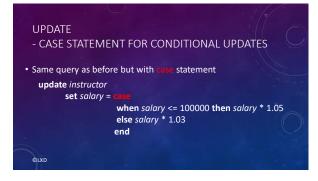


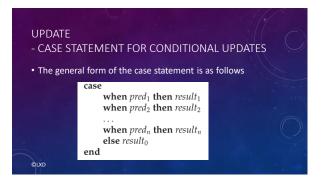












• Recompute and update tot\_creds value for all students

update student S

set tot\_cred = (select sum(credits)
from takes, course
where takes.course\_id = course.course\_id and
S.ID= takes.ID and takes.grade <> 'F' and
takes.grade is not null);

• Sets tot\_creds to null for students who have not taken any course

