



### **Integrity Constraints in Create Table**

- primary key  $(A_1, ..., A_n)$ check(P)
- unique(A<sub>j1</sub>, A<sub>j2</sub>, ..., A<sub>jm</sub>)

Example: Declare branch\_name as the primary key for branch and ensure that the values of assets are non-negative.

create table branch (branch\_name char(15),

branch\_city char(30), assets integer, primary key (branch\_name), check (assets>=0))

primary key declaration on an attribute automatically ensures **not null** in SQL-92 onwards, needs to be explicitly stated in SQL-89



# **Drop and Alter Table Constructs**

- The drop table command deletes all information about the dropped relation from the database.
- The alter table command is used to add attributes to an existing

#### alter table r add A D

where A is the name of the attribute to be added to relation r and Dis the domain of A.

- All tuples in the relation are assigned null as the value for the new attribute.
- The alter table command can also be used to drop attributes of a relation:

#### alter table r drop A

where A is the name of an attribute of relation r

• Dropping of attributes not supported by many databases



em Concepts - 5th Edition, June 15, 2005



# **Basic Query Structure**

- SQL is based on set and relational operations with certain modifications and enhancements
- A typical SQL query has the form:

 $\begin{array}{l} \textbf{select} \ A_1, \ A_2, \ ..., \ A_n \\ \textbf{from} \ r_1, \ r_2, \ ..., \ r_m \\ \textbf{where} \ P \end{array}$ 

- A<sub>i</sub>represents an attribute
- R<sub>i</sub> represents a relation
- P is a predicate.
- This query is equivalent to the relational algebra expression.

$$\prod_{A_1,A_2,\ldots,A_n} (\sigma_P(r_1 \times r_2 \times \ldots \times r_m))$$

■ The result of an SQL query is a relation.



#### The select Clause

- The select clause list the attributes desired in the result of a query
  - corresponds to the projection operation of the relational algebra
- Example: find the names of all branches in the *loan* relation:

select branch\_name from loan

In the relational algebra, the query would be:

 $\Pi_{\mathit{branch\_name}}$  (loan)

- NOTE: SQL names are case insensitive (i.e., you may use upper- or lower-case letters.)
  - . Some people use upper case wherever we use bold font.





## The select Clause (Cont.)

- SQL allows duplicates in relations as well as in query results.
- To force the elimination of duplicates, insert the keyword distinct after
- Find the names of all branches in the *loan* relations, and remove duplicates

select distinct branch\_name from loan

■ The keyword all specifies that duplicates not be removed.

select all branch\_name from loan



## The select Clause (Cont.)

An asterisk in the select clause denotes "all attributes"

select '

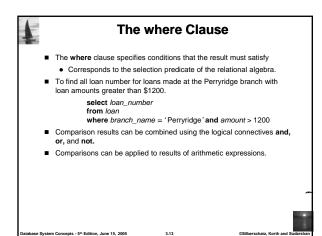
from loan

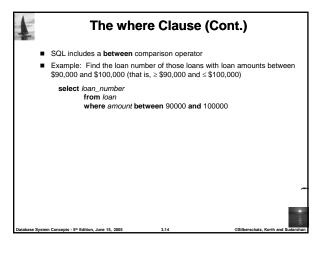
- The select clause can contain arithmetic expressions involving the operation, +, -, \*, and /, and operating on constants or attributes of
- The query:

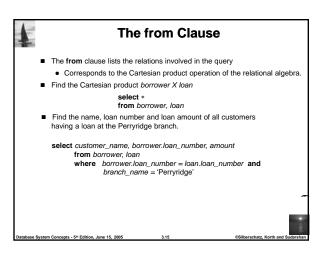
select loan\_number, branch\_name, amount \* 100

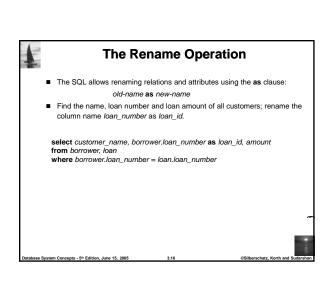
would return a relation that is the same as the loan relation, except that

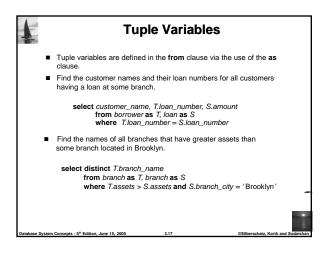
the value of the attribute amount is multiplied by 100.

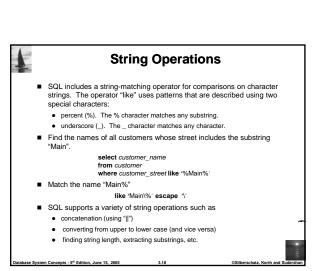














## **Ordering the Display of Tuples**

■ List in alphabetic order the names of all customers having a loan in Perryridge branch

select distinct customer\_name

from borrower, loan

where borrower loan number = loan.loan number and branch\_name = 'Perryridge'

order by customer\_name

 $\blacksquare$  We may specify  $\mbox{\bf desc}$  for descending order or  $\mbox{\bf asc}$  for ascending order, for each attribute; ascending order is the default.

• Example: order by customer\_name desc



## **Duplicates**

- In relations with duplicates, SQL can define how many copies of tuples appear in the result.
- Multiset versions of some of the relational algebra operators given multiset relations  $r_1$  and  $r_2$ :
  - 1.  $\sigma_{\theta}(r_1)$ : If there are  $c_1$  copies of tuple  $t_1$  in  $r_1$ , and  $t_1$  satisfies selections  $\sigma_{\theta}$ , then there are  $c_1$  copies of  $t_1$  in  $\sigma_{\theta}(r_1)$ .
  - 2.  $\Pi_A(r)$ : For each copy of tuple  $t_1$  in  $r_1$ , there is a copy of tuple  $\Pi_A(t_{\rm 1}) \ln \Pi_A(r_{\rm 1})$  where  $\Pi_A(t_{\rm 1})$  denotes the projection of the single
  - 3.  $r_1 \times r_2$ : If there are  $c_1$  copies of tuple  $t_1$  in  $r_1$  and  $c_2$  copies of tuple  $t_2$  in  $r_2$ , there are  $c_1 \times c_2$  copies of the tuple  $t_1$ .  $t_2$  in  $r_1 \times r_2$



## **Duplicates (Cont.)**

■ Example: Suppose multiset relations  $r_1$  (A, B) and  $r_2$  (C) are as

$$r_1 = \{(1,\ a)\ (2,a)\} \qquad r_2 = \{(2),\ (3),\ (3)\}$$

- Then  $\Pi_B(r_1)$  would be {(a), (a)}, while  $\Pi_B(r_1)$  x  $r_2$  would be  $\{(a,2), (a,2), (a,3), (a,3), (a,3), (a,3)\}$
- SQL duplicate semantics:

select  $A_1, A_2, ..., A_n$  $\begin{array}{lll} \text{from} \ r_{\text{1}}, \ r_{\text{2}}, \ ..., \ r_{m} \\ \text{where} \ P \end{array}$ 

is equivalent to the multiset version of the expression:

$$\prod_{A_1,A_2,...,A_n} (\sigma_P(r_1 \times r_2 \times ... \times r_m))$$



## **Set Operations**

- The set operations union, intersect, and except operate on relations and correspond to the relational algebra operations  $\cup$ ,  $\cap$ ,
- Each of the above operations automatically eliminates duplicates; to retain all duplicates use the corresponding multiset versions union all, intersect all and except all.

Suppose a tuple occurs m times in r and n times in s, then, it occurs:

- m + n times in r union all s
- min(m,n) times in r intersect all s
- max(0, m − n) times in r except all s



#### **Set Operations**

■ Find all customers who have a loan, an account, or both:

(select customer\_name from depositor)

union

(select customer\_name from borrower)

■ Find all customers who have both a loan and an account.

(select customer\_name from depositor)

(select customer\_name from borrower)

■ Find all customers who have an account but no loan.

(select customer\_name from depositor)

except (select customer\_name from borrower)



## **Aggregate Functions**

■ These functions operate on the multiset of values of a column of a relation, and return a value

avg: average value min: minimum value max: maximum value sum: sum of values count: number of values

