More SQL

CS157A

Chris Pollett

Oct. 17, 2005.

Outline

- Specifying Key and Referential Integrity Constraints
- Naming Constraints
- Drop, Alter
- Basic Queries in SQL

Specifying Key Constraints

- We have already seen we can indicate a primary key in a create table statement using the syntax: PRIMARY KEY(COL1, COL2,…)
- If one has a single attribute key one can mark it so, when one declares the column type. For instance,

  DNUMBER INT PRIMARY KEY,
- If one wants to indicate secondary keys, one can use the syntax UNIQUE(COL1, COL2, …)

Referential Integrity Constraints

- Consider the following line from a create table statement which specifies a foreign key constraint:

  FOREIGN KEY (DNUMBER) REFERENCES DEPARTMENT(DNUMBER), ON DELETE CASCADE ON UPDATE CASCADE

- ON DELETE and ON UPDATE are used to specify **referential triggered actions**.
- If ON DELETE and ON UPDATE had not been specified then by default deleting a DEPARTMENT DNUMBER which is referenced by the above would result in the operation being rejected.
- With ON DELETE CASCADE, we are saying if a DEPARTMENT DNUMBER is deleted then anything in this table that references it should be deleted.
- With ON UPDATE CASCADE, we are saying if a DEPARTMENT DNUMBER is changed then relevant rows in this table should be changed to the new DNUMBER.
- Besides CASCADE, one could also use other actions such as SET NULL or SET DEFAULT.

Naming Constraints

- If one ever has to change ones table, it is useful to have a name for ones constraints.
- That way, one can easily modify or drop the constraint if one has to.
- An example of naming a constraint would be the following code fragment:
- CONSTRAINT DEPTPK PRIMARY KEY(DNUMBER)
- DEPTPK can now be used to refer to this constraint.

Drop

- The DROP command can be used to drop schemas as well as named schema elements, such as tables, domains, or constraints.
- Here are some examples:

  DROP SCHEMA COMPANY;

  DROP SCHEMA COMPANY RESTRICT;

  DROP TABLE DEPENDENT;

  DROP TABLE DEPENDENT CASCADE;
- RESTRICT prevents the operation unless all the things that are in COMPANY have been deleted.
- CASCADE drops any constraints that reference the table as well.

Alter

- To change the definition of a table one uses the alter command:

```
ALTER TABLE COMPANY.EMPLOYEE ADD JOB
    VARCHAR(12); /* add a column job */
ALTER TABLE COMPANY.EMPLOYEE ALTER JOB
    SET DEFAULT ‘GRADING’; /* alter job */
ALTER TABLE COMPANY.EMPLOYEE DROP JOB
    CASCADE; /* drop job */
ALTER TABLE DEPARTMENT DROP CONSTRAINT
    DEPTPK; /* drop a constraint notice this time we are
    assuming DEPARTMENT is in the current schema */
```

Basic Queries in SQL

- The basic format of a query in SQL is

  SELECT <attribute list> -- this line is like a projection

  FROM <table list> -- this line is like a cartesian product

  WHERE <condition>; -- this line is like a selection

- For example, to do the query:
  - Retrieve the birthdate and address of the employee(s) whose name is ‘John B. Smith’.
  - One could do:

```
SELECT BDATE, ADDRESS
FROM EMPLOYEE
WHERE FNAME=‘John’ AND MINIT=‘B’ AND
   LNAME=‘Smith’;
```

Some More Examples

- Retrieve the name and address of all employees who work for the ‘research’ department.

  SELECT FNAME, LNAME, ADDRESS

  FROM EMPLOYEE, DEPARTMENT

  WHERE DNAME=‘Research’ and DNUMBER=DNO;

- For every project located in ‘Stafford’ list the project number, the controlling department numbers, and the department manager’s last name, address, and birthdate.

  SELECT PNUMBER, DNUM, LNAME, ADDRESS, BDATE

  FROM PROJECT, DEPARTMENT, EMPLOYEE

  WHERE DNUM=DNUMBER AND MGRSSN=SSN and PLOCATION=‘Stafford’;

Ambiguous Attribute Names, Aliasing, and Tuple Variables

- What do you do if you have two tables with the column A?

  SELECT T1.A, T2.A

  FROM T1, T2;

- How do you join a table with itself? For instance, suppose for each employee, you want to retrieve the employee first and last name and his manager's first and last name.

  SELECT E.FNAME, E.LNAME, S.FNAME, S.LNAME

  FROM EMPLOYEE AS E, EMPLOYEE AS S

  WHERE E.SUPERSSN= S.SSN;

- E and S are called *aliases* or *tuple variables*.
- It is also possible to create aliases for both tables and columns:

  EMPLOYEE AS E(FN, MI, LN, SSN, BD, ADDR, SEX, SAL, SSSN, DNO)

Unspecified WHERE clause and Use of the Asterisk

- If a WHERE clause is not specified then it indicates that no condition needs to hold on the tuples, so any tuples in the from clause will work.
- For instance, SELECT SSN,DNAME FROM EMPLOYEE, DEPARTMENT; returns all combinations of SSN from EMPLOYEE and DNAME from department.
- If we want to return all columns that could result from a query we use a * in the select line:

```
SELECT * FROM EMPLOYEE WHERE DNO=5;
SELECT * FROM EMPLOYEE, DEPARTMENT;
    /* this last is like a cartesian product */
```

Tables as Sets in SQL

- Usually SQL treats query results as multisets.
- So SELECT SALARY FROM EMPLOYEE; might return several identical salaries. Note: to emphasize we want a multiset we can write: SELECT ALL SALARY FROM EMPLOYEE;
- If one wants the result as a set one can use the keyword distinct; SELECT DISTINCT SALARY FROM EMPLOYEE;
- SQL also supports set union (UNION), set difference (EXCEPT), and set intersection (INTERSECT). For example,

  (SELECT DISTINCT SALARY FROM EMPLOYEE WHERE DNO=4)

  UNION

  (SELECT DISTINCT SALARY FROM EMPLOYEE WHERE DNO=6)
- There are also multiset analogs of these operations UNION ALL, EXCEPT ALL, and INTERSECT ALL.

Substring Pattern Matching and Arithmetic Operators

- Pattern matching in SQL can be done using the keyword LIKE. The symbol % is used to replaces zero or more characters and _ is used to match any one character. For example,

  SELECT FNAME, LNAME FROM EMPLOYEE
  WHERE ADDRESS LIKE '%Ho_ston, TX%';

- Arithmetical operators can be applied to the outputs of a query:

  SELECT FNAME, LNAME, 1.1*SALARY AS
  INCREASED_SAL FROM EMPLOYEE;

BETWEEN and ORDER BY

- BETWEEN is a useful keyword for specifying a domain in a where clause:

  SELECT * FROM EMPLOYEE WHERE (SALARY BETWEEN 50000 AND 60000);

- Another useful operation to do is to be able to sort and order the results of a query. An SQL ORDER BY clause is used to do this:

  SELECT LNAME, FNAME, BDATE FROM EMPLOYEE ORDER BY LNAME DESC, FNAME DESC, BDATE ASC;

  – If don't say ascending or descending then by default ascending.