The Relational Model

CS157A

Chris Pollett

Sept. 19, 2005.

Outline

- A little bit on Oracle on sigma
- Introduction to the Relational Model

Oracle on Sigma

- Two ways to connect:
  - connect to sigma, then connect to the DBMS
  - connect directly to DBMS

Method I

- ssh username@sigma.cs.sjsu.edu
- (Off campus you might have to ssh to your cs110L account and then go from there to sigma)
- At the unix prompt type source /cs157aenv
- Then type sqlplus.
- You will be prompted for you Oracle password and login.

Method II

- Use sqlplus on your home machine:

  sqlplus username@connect_identifier
- You need to set up a connect_identifier e for this to work.

One way to set up a service

- Drill down $ORACLE_HOME --> network --> admin. Open file tnsnames.ora in a text editor. Add an entry like:

```
CS157A =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST =
    sigma.cs.sjsu.edu)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = cs157a.cs.sjsu.edu)
    )
  )
```

- CS157A will be your connect identifier.

Some Basic SQLPLUS commands

EXIT -- leave sqlplus

@filename -- execute a sequence of SQL commands in a file.

HELP *command* -- get help for syntax of a command

HOST - execute OS command.

SPOOL *filename* -- start sending everything typed to a file. (SPOOL OFF to stop).

EDIT -- load contents of buffer into editor to edit. Very useful if make typos

SET -- used to set an oracle variable. Example: set echo on

SHOW ALL - show all variables

DESCRIBE table -- show columns of a table

DEFINE _editor=pico -- set the default editor.

Buffer Manipulation

- Commands are stored in a buffer which can be altered to correct for typos.
- To see the contents of the buffer can type **list** or **list** *n m*.
- Typing a number<ret> at the prompt will go to that line of the buffer.

  SQL> 5

  5* where streets='no_name';
- append *text* -- adds text as last line of buffer.
- change /*old*/*new* -- changes old to new on current line of buffer
- del -- deletes current line
- clear buffer -- deletes contents of buffer.
- save *file* -- saves buffer contents to file.

Introduction to the Relational Model

- So far we have been working with Conceptual models (i.e., ER diagrams) to figure out how to organize our data in the database.
- We are now going to look at the most popular representational model: the Relational Model.
- It was developed by Codd in 1970.

Relational Model Concepts

- In the relational model, a database is viewed as a collection of *relations*.
- A relation can be thought of as a table of values:

Student	Name	StudentNumber	Class	Major
	Bob	123456789	Jr	CS
	Sally	912345678	Sr	PolySci

Parts of a Relation

- Notice in our relation on the previous slide the relation has a name, and the column headings have names.
- The column names are called *attributes*.
- The space of possible values for entries under a given column are called the attribute's *domain*.
  - For example, StudentNumber might be an INTEGER.
- A row out of our relation is called a *tuple*.

Relational Schemas

- A *relational schema R* is used to specify the format of a relation. It is made up of a relation name followed by a list of attributes. For example, $R(A_1,\ldots,A_n)$.
- We denote the domain of each A_i by $dom(A_i)$.
- We will assume a value in a domain is always *atomic* as far as the database goes. That is, indivisible.
- The degree of a relation R is the number of attributes in it. For above case, we have n.
- Using this notation our STUDENT example above would be written:

  STUDENT(Name: VARCHAR(20), Student Number: INTEGER,
  Class :{Fr, So, Jr, Sr}, Major : CHAR(2))

Relations

- We can now talk about a *relation* r(R) (aka a *relation state*) for a given relational schema $R(A_1,\ldots,A_n)$.
- r(R) is a set of tuples $\{t_1,.. t_m\}$ such that each t_i belongs to

  $dom(A_1) \times \ldots \times dom(A_n)$.

Characteristics of Relations

We now discuss some properties of relations that make them independent of how the data is physically stored.

1. Ordering of tuples in a Relation does not matter. i.e., relations are sets of tuples not lists of tuples.
2. Can modify our definition above so that ordering of attributes within a tuple does not matter as well.
3. Values and nulls in tuples. Atrributes are assumed to be atomic, not composite or multivalued. A special value null is used when the attribute does not apply.