CS 580 Client-Server Programming
Fall Semester, 2002
Doc 21 SQL Part 1
Contents

References

PostgreSQL Developer's Handbook, Geschwinde, Schönig, Sams, 2002

PostgreSQL Interactive Documentation http://www.postgresql.org/idocs/

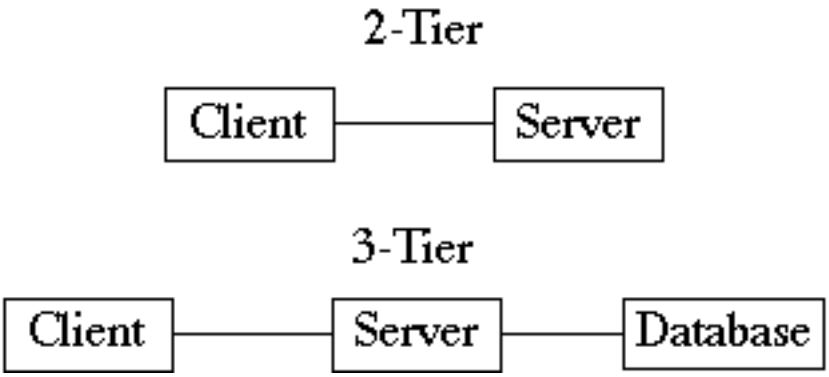
PostgreSQL Technical Documentation Web site, http://techdocs.postgresql.org/

Andrew Scherpbier's CS580 Lecture notes http://www.eli.sdsu.edu/courses/spring97/cs596/notes/databases/databases.html

Database

Most servers will use some sort of database.

Jargon



What is a database?

A database holds information and provides for a mechanism to access this information.

Examples of some common (electronic) databases:

- Unix password file
- IRS records system
- Rolodex(TM)
- Computer file system
- Library (object files)
- Student grades
- Telephone directory

Database terms

Table:

- A collection of records

Record:

- A collection of fields which belong together

Field:

- A piece of data (datum)

Example record:

Lastname Firstname MI ID Phone Email

Example table of these records:

Scherpbier	Andrew	W	42	45026	none
Over	Ben	D	57	411	a@b

Jargon

Sometimes **database** means a program for managing data

Oracle Corporation is a database company.
MS Access is database.

Sometimes **database** means a collection of data

I keep a database of my CD collection on 3 by 5 cards

Sometimes **database** means a set of tables, indexes, and views

My program needs to connect to the Airline Reservation database, which uses Oracle

Some Common Issues

Indexing

We need a reasonably fast way to get at the data.

Indexing provides a method to find records.

An index maps keys to records

Keys can be:

- One or more fields of a record
- Independent from a record

Keys are normally stored in a separate data structure.
Some common data structures used:

- Sorted for binary search
- Tree (btree or some other efficient tree structure)
- Hash table

Multiple indexes can be created on a single table

Concurrency

Goal:

- Allow multiple access to data and keep data consistent

Some solutions:

- Record locking (complex)
- Table locking (easier...)
- Access serialization (frequently not possible)
- Allow only one "updater" per table (One writer, Many readers)

Other problems:

- Caching (for efficiency)
- Permission (user rights)

Roll your own vs. using existing

Many database management systems are available

For simple database requirements can implement your own database

Types of Databases

Relational

Data is stored in tables

Object-Oriented

Tables can be subclassed

Programmer can define methods on tables

Object

Objects are stored in the database

Relational, Object-Oriented Databases and SQL

Database consists of a number of tables

Table is a collection of records

Each Column of data has a type

firstname	lastname	phone	code
John	Smith	555-9876	2000
Ben	Oker	555-1212	9500
Mary	Jones	555-3412	9900

Use Structured query language (SQL) to access data

Some Available Databases

Oracle
DB2
SQL Server
Access
MySQL
PostgresSQL
Informix
Ingres
InterBase
Sybase
FileMaker Pro
FoxPro
Paradox
dBase

SQL History

Structured query language (SQL)

Dr. E. F. Codd develops relational database model
- Early 1970's

IBM System R relational database
- Mid 1970's
- Contained the original SQL language

First commercial database - Oracle 1979

SQL was aimed at:
- Accountants
- Business people

SQL89
- Not well followed
- ANSI X3.135-1989

SQL92
- First commonly followed standard
- ANSI X3.135-1992
- SQL2

ISO/IEC 9075-1 through 5
- New SQL standard

PostgreSQL Names

Databases, tables columns & indexes have names

Legal Characters

Alphanumeric characters
'_'
'$'

Names can start with:
Letter
Underscore
Letter with diacritical marks and some non-latin letters

Name length

Only the first 31 characters of names are used for:

- tables, databases, columns & indexes

Names are not case sensitive

PostgreSQL Data Types

- Numeric Values
  Integer - decimal or hex
  Floating-point - scientific & 12.1234

- String Values
  Use single quotes
  ‘this is a string’

Sequence	Meaning
\'	Single quote
\b	Backspace
\n	Newline
\r	Tab
\\	Backslash
\xxxx	Character were xxxx is the octal of ASCII code

Including a quote character in a string
Double quote the character
'Don''t do it'

Escape the quote character with a backslash
'Don\'t do it'

Comments

-- this is a comment

/* this is also a comment */

Numeric Data Types

Type name	Description	Range
`smallint`	Fixed-precision	-32768 to +32767
`integer`	Usual choice for fixed-precision	-2147483648 to +2147483647
`bigint`	Very large range fixed-precision	-9223372036854775808 to 9223372036854775807
`decimal`	user-specified precision, exact	no limit
`numeric`	user-specified precision, exact	no limit
`real`	variable-precision, inexact	6 decimal digits precision
`double precision`	variable-precision, inexact	15 decimal digits precision
`serial`	autoincrementing integer	1 to 2147483647
`bigserial`	autoincrementing integer	1 to 9223372036854775807

Numeric(10, 2) defines a number with maximum of 10 digits with 2 of the 10 to the right of the decimal point

12345678.91

decimal and numeric are different names for the same type

String Types

Type	Description
character(*n*), char(*n*)	Fixed-length blank padded
character varying(*n*), varchar(*n*)	Variable-length with limit
text	Variable unlimited length
bytea	Variable (not specifically limited) length binary string

CHAR & VARCHAR are the most common string types

CHAR is fixed-width

Shorter strings are padded

TEXT can be any size

Some databases use BLOB instead of bytea

PostgreSQL limits strings to 1GB in storage space

Date & Time Types

Type	Description
`timestamp [(`*p*`)] without time zone`	both date and time
`timestamp [ (`*p*`) ] [ with time zone ]`	both date and time
`interval [ (`*p*`) ]`	for time intervals
`date`	dates only
`time [ (`*p*`) ] [ without time zone ]`	times of day only
`time [ (`*p*`) ] with time zone`	times of day only

(p) indicates optional number of fractional digits retained in the seconds field

Date Formats

Example	Description
January 8, 1999	Unambiguous
1999-01-08	ISO-8601 format, preferred
1/8/1999	U.S.; read as August 1 in European mode
8/1/1999	European; read as August 1 in U.S. mode
1/18/1999	U.S.; read as January 18 in any mode
19990108	ISO-8601 year, month, day
990108	ISO-8601 year, month, day
1999.008	Year and day of year
99008	Year and day of year
J2451187	Julian day
January 8, 99 BC	Year 99 before the Common Era

Setting the Date Format

SET DateStyle TO ‘US’
SET DateStyle TO ‘NonEuropean’

Sets date format to month day year

SET DateStyle TO ‘European’

Sets date format to day month year

Default is ISO style

Common SQL Statements

SELECT	Retrieves data from table(s)
INSERT	Adds row(s) to a table
UPDATE	Changes field(s) in record(s)
DELETE	Removes row(s) from a table Data Definition
CREATE TABLE	Define a table and its columns(fields)
DROP TABLE	Deletes a table
ALTER TABLE	Adds a new column, add/drop primary key
CREATE INDEX	Create an index
DROP INDEX	Deletes an index
CREATE VIEW	Define a logical table from other table(s)/view(s)
DROP VIEW	Deletes a view

SQL is not case sensitive

SQL: CREATE

Creates a table.

```
CREATE TABLE table_name (
   col_name    col_type [ NOT NULL | PRIMARY KEY]
   [, col_name col_type [ NOT NULL | PRIMARY KEY]]*
)
```

Some Examples

Examples types in psql – interactive PostgreSQL terminal

```
CREATE TABLE students
(
   firstname  CHAR(20) NOT NULL,
   lastname   CHAR(20),
   phone      CHAR(10),
   code       INTEGER
);

CREATE TABLE codes
(
   code       INTEGER,
   name       CHAR(20)
)


SELECT * FROM students;

 firstname | lastname | phone | code
-----------+----------+-------+------
(0 rows)
```

Inserting

```
INSERT
   INTO students (firstname, lastname, phone, code)
   VALUES ('Roger', 'Whitney', '594-3535', 2000 );

INSERT
   INTO codes (code, name)
   VALUES (2000, 'marginal' );

SELECT * FROM students;

     firstname      |      lastname      |   phone    | code
--------------------+--------------------+------------+------
 Roger              | Whitney            | 594-3535   | 2000
(1 row)

SELECT students.lastname, codes.name
WHERE students.code=codes.code;

      lastname      |        name
--------------------+--------------------
 Whitney            | marginal
(1 row)
```