PSUG National Information Exchange

Users helping users

Basic SQL

Dean Dahlvang
Proctor Public Schools
Proctor, MN

About Dean

- Dean Dahlvang (ddahlvan@proctor.k12.mn.us)
- Director of Administrative Technology for the Proctor Public Schools (just outside of Duluth, Minnesota at the tip of Lake Superior).
- PowerSchool Admin for 10 years, PowerSchool contracted state reporting programmer for the last 5 years.
- I enjoy winter camping in the BWCA.

Basic SQL


WELCOME
TO Fabulous
2011 PSUG NATIONAL
Information Exchange

What is SQL?

- SQL stands for Structured Query Language
- It is a standardized programming language that allows for quick access to a relational database management system (rdbms).
- S Q L or “Sequel” can update, delete, create and drop data…but it is best known for it’s ability to SELECT data.

Knowing Your Data

- Before we can use SQL, it is important to have a good idea of what kind of data we have.
- Data Models or Entity Relations Diagrams refer to how data exists in a rdbms.
- Inside a database, data lives in tables. Inside tables, are rows (or records) that contain columns (or fields).

Tables and Columns and Rows Oh My!

- Tables – Students, Reenrollments, CC, Teachers
- Columns – StudentId, Course_Name, Grade_Level
- Rows – Exist with a table that contain specific data inside its columns.
- PowerSource has a data dictionary…just search for it.

How Do We Get The Data?

- SQL Developer is a tool (query browser or query analyzer) that comes with Oracle…the database (rdbms) that holds the PowerSchool data.
- Launch SQL Developer

Hello World!

- Two basics in the SELECT statement – SELECT and FROM
- SELECT columns FROM tables
- SELECT lastfirst, email_addr FROM teachers
- See why it is important to know your data?
- What else can we select from the teachers table?

Try Some!

- SELECT lastfirst, grade_level, gender FROM students
- SELECT course_number, course_name FROM courses
- SELECT yearid, name, schoolid FROM terms
- SELECT * FROM teachers

Filtering

- We can add a clause (no, not Santa) to our SELECT statement that will filter the data.
- WHERE clause allows us to specify that certain criteria are met for the data returned from our database
- SELECT lastfirst, grade_level FROM students WHERE enroll_status=0

Queries…Now with Extra Filtering!

- SELECT course_number, course_name FROM courses WHERE schoolid=100
- SELECT yearid, name, schoolid FROM terms WHERE portion=1
- SELECT * FROM teachers WHERE status=1

Where Clause Wonderment

- The Where clause can get pretty fancy. Beyond normal “equality”, it can do a number of logical operations and functions.
- SELECT lastfirst, grade_level FROM students WHERE enroll_status=0 and grade_level=12
- …WHERE enroll_status=0 and grade_level in (9,12)
- …WHERE enroll_status=0 and last_name=‘Smith’
- …WHERE enroll_status=0 and in (‘Jones’,’Smith’)
- …WHERE enroll_status=0 and first_name like ‘Jo%’

Order in the Court

- The Order By clause of a Select statement will allow for the sorting of data. Multiple fields in either direction can be sorted
- SELECT lastfirst, grade_level FROM students WHERE enroll_status=0 …
- …ORDER BY grade_level
- …ORDER BY grade_level DESC
- …ORDER BY grade_level DESC, lastfirst ASC

Beyond a Single Table

- It is important to remember how tables and data relate within the database. Multiple tables can be accessed within a single Select statement as long as there is a common relationship between the tables.
- The heinous Cartesian Product (insert sneer)
- Remember the importance of the data model

The Join

- The Join clause allows us to take advantage of pulling data from multiple tables (assuming they have a relationship.)
- Look at two related tables:
  - Students
  - Schools
  - How are these tables related? Students.id vs Schools.studentid?
  - A quick word about convention

The Join Example

- SELECT lastfirst, grade_level, schoolid, abbreviation
  FROM students
  INNER JOIN schools ON students.schoolid = schools.school_number
  WHERE enroll_status=0
  ORDER BY lastfirst
- What happens if my tables share the same column name? Our fix is to ALIAS the tables and columns. Lets try this again:
- SELECT s.lastfirst, s.grade_level, s.schoolid, sch.abbreviation
  FROM students s
  INNER JOIN schools sch ON s.schoolid = sch.school_number
  WHERE s.enroll_status=0
  ORDER BY s.lastfirst

Flyers and Bombers

- This example goes well beyond a lesson in World War II History. Let's talk about the students who failed classes first semester!
- Suppose we want a list of students and the classes they failed for first semester…what is our approach?
- Table or Tables? If more than one, how do they relate?

Flyers and Bombers part 2

- SELECT * FROM storedgrades sg WHERE sg.termid=2001 and sg.grade='F'
- If we limit our column selection, what should they be?
- Studentid isn't easy to use, how do we add the student's name?

Flyers and Bombers part 3

- SELECT s.lastfirst, sg.storecode, sg.grade, sg.absences, sg.tardies, sg.course_name FROM storedgrades sg INNER JOIN students s ON s.id=sg.studentid WHERE sg.termid=2001 AND sg.grade='F’ ORDER BY s.lastfirst

Flyers and Bombers part 4

- Include number of students in the classes by joining the Sections table

- SELECT s.lastfirst, sg.storecode, sg.grade, sg.absences, sg.tardies, sg.course_name, sec.no_of_students FROM storedgrades sg INNER JOIN students s ON s.id=sg.studentid INNER JOIN sections sec ON sec.id=sg.sectionid WHERE sg.termid=2001 AND sg.grade='F‘ ORDER BY s.lastfirst

Let's Look at a Matt Example

- Matt Freund has a great example:
- SELECT att_date, schoolid, attendance_codeid
  FROM attendance
  WHERE yearid=20 and
  att_mode_code='ATT_ModeMeeting' and
  studentid=3
  ORDER BY att_date

Matt Example part 2

- To make it more meaningful, we can join the schools and attendance_code tables to the attendance table and replace schoolid with abbreviation, and attendance_codeid with description:
- SELECT att.att_date, sch.abbreviation, ac.description
  FROM attendance att
  INNER JOIN schools sch on att.schoolid = sch.school_number
  INNER JOIN attendance_code ac ON att.attendance_codeid = ac.id
  WHERE att.yearid=20 and att.att_mode_code='ATT_ModeMeeting' and att.studentid=3
  ORDER BY att.att_date

Matt Example part 3

- Let's add one more item to the output – the course name. Adding the course name gives a good example of the power of SQL. The course name is held in the courses table, however, there is no connection between the attendance and courses tables. But there is a connection between attendance and the CC table, so we can join those two, and there's a connection between CC and courses, so we can join those two, and then pull information from the courses table:

Matt Example part 3 Code

- SELECT att.att_date, sch.abbreviation, ac.description, c.course_name
  FROM attendance att
  INNER JOIN schools sch ON att.schoolid = sch.school_number
  INNER JOIN attendance_code ac ON att.attendance_codeid = ac.id
  INNER JOIN cc cc ON att.ccid = cc.id
  INNER JOIN courses c on cc.course_number = c.course_number
  WHERE att.yearid=20 and att.att_mode_code='ATT_ModeMeeting' and att.studentid=3
  ORDER BY att.att_date

- If the date looks a little fishy, it can be reformatted to a string:
  - To_char(att.att_date,'MM/DD/YYYY')

Using the Queries – Tlist_sql

- After creating a query, how do you make it available for others to use? Aha! Tlist_sql
- Tlist_sql will allow us to embed a sql select statement inside a web table.
- Here is a simple staff directory:
  - SELECT t.lastfirst, t.email_addr, t.school_phone FROM teachers t WHERE schoolid=100 AND status=1

Tlist_sql Sample Code

```
<table border="0" cellspacing="0" cellpadding="4" width="100%">
<tr>
<td class="bold">Name</td>
<td class="bold">Email</td>
<td class="bold">Phone</td>
</tr>
<tr>
~[tlist_sql;SELECT t.lastfirst, t.email_addr, t.school_phone FROM teachers t WHERE
    schoolid=100 AND status=1;nonemessage=No courses found.]
<td>~(t.lastfirst)</a></td>
<td>~(t.email_addr)</td>
<td>~(t.school_phone)</td>
</tr>
[/tlist_sql]
</table>
```

Using the Queries - sqlReports

- sqlReports is a simple way of creating sql queries inside of PowerSchool without the need to have a traditional sql client and odbc access. Both of those things definitely help however.
- It allows you to create queries that are available to end users to execute.
- It is like a fancy List Students function only you can query any table or view, save the query for repeated use.
- sqlReports can be downloaded from powerdatasolutions.org

Some Aggregate Functions

- Quick examples of some aggregate functions
- Distinct
  - SELECT DISTINCT city FROM students
- Count
  - SELECT city, count(*) FROM students GROUP BY city ORDER BY city
- Sum
  - SELECT grade_level, sum(balance1) FROM students GROUP BY grade_level

Questions?


WELCOME
TO Fabulous
2011 PSUG NATIONAL
Information Exchange