CSC 380/530 — Advanced Database
Project 3 (document version 1.0)
Data Warehousing Concepts using SQL and PL/SQL

- This project is due by 11:59:59 PM on Monday, December 7, 2015. Projects are to be submitted electronically.
- This project will count as 25% of your final course grade.
- This project is to be completed **individually**. Do not share your work with anyone else.

Overview

The focus of this course thus far has been on transactional processing via SQL and PL/SQL. More generally, the focus of such database design and implementation is often on designing and tuning inserts, updates, and queries/selects. This encompasses what is known as On-Line Transaction Processing (OLTP).

In this last project, we will instead focus on On-Line Analytical Processing (OLAP), which centers around back-end analytical reporting.

OLTP versus OLAP

OLTP focuses on operational (i.e., day-to-day) transactions. Because of this, back-end reporting is often difficult, nonexistent, or extremely slow. OLTP systems are typically business process-driven systems handling current data that is regularly updated (e.g., course registration systems, banking systems, etc.). Such systems are generally designed and tuned for fast inserts and updates on relatively small datasets.

An OLAP system (e.g., a data warehouse) focuses on efficient back-end analytical reporting in support of enterprise-wide decision-making processes for an organization, often all the way up to the executive level.

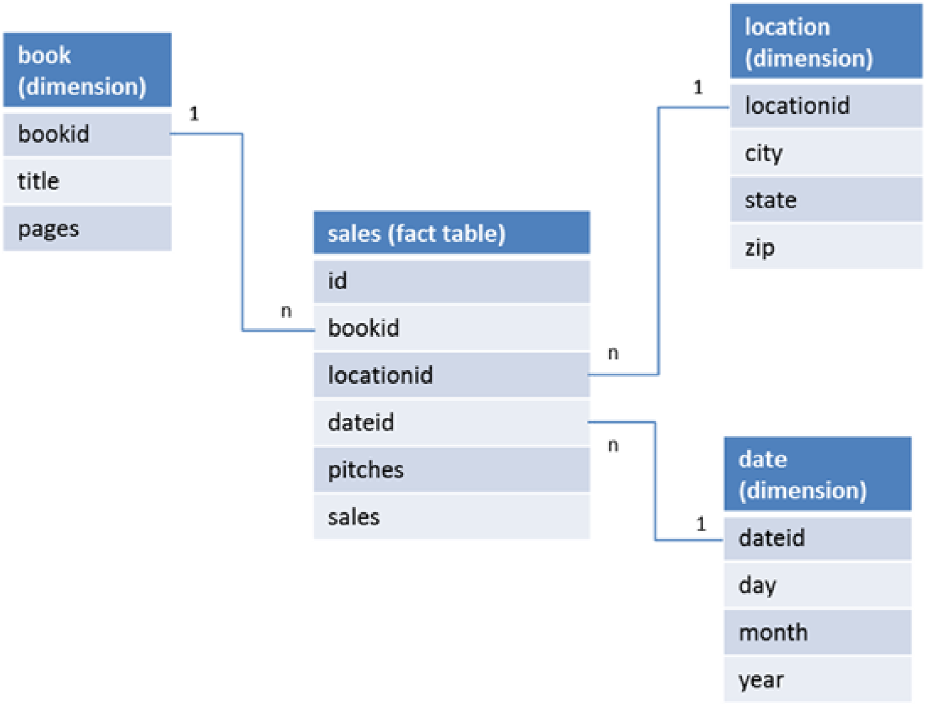
To achieve this, OLAP systems provide long-term historical data that users may filter based on a broad array of attributes called “dimensions.” Such analysis is called “slicing and dicing” the data.

OLTP and OLAP systems generally work side by side, with the OLAP system extracting transactional data from the OLTP system on a regular basis (e.g., nightly). Data within an OLAP system is read-only, historical, and generally never goes away.

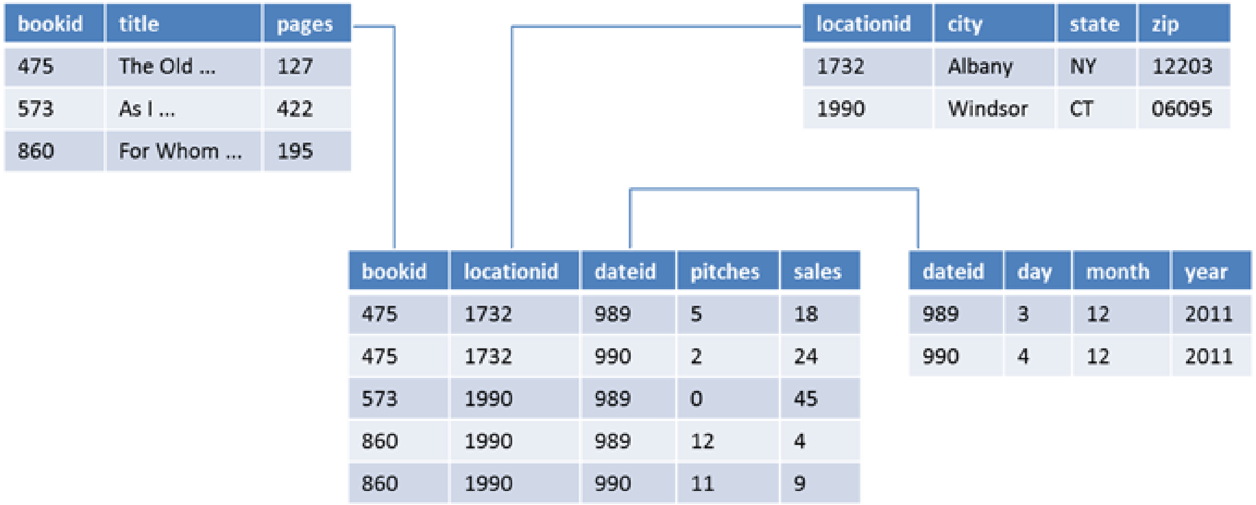
Part of the challenge of data warehouse design is to ensure fast query times over large volumes of data. To accomplish this, a common approach is the use of the “star schema,” which consists of a “fact table” surrounded by “dimension tables.”

A fact table consists of a number of foreign keys and (typically) a number of additive numeric values. The foreign keys correspond to primary keys within each of the dimension tables.

As an example, below is a star schema with fictional sales data:



A row in the `sales` fact table holds “pitches” and “sales” counts for one particular combination of book, location, and date. Example data is shown below:



bookid	title	pages
475	The Old ...	127
573	As I ...	422
860	For Whom ...	195

locationid	city	state	zip
1732	Albany	NY	12203
1990	Windsor	CT	06095

bookid	locationid	dateid	pitches	sales
475	1732	989	5	18
475	1732	990	2	24
573	1990	989	0	45
860	1990	989	12	4
860	1990	990	11	9

dateid	day	month	year
989	3	12	2011
990	4	12	2011

Given the schema and sample data shown above, write SQL queries as described below.

1. How many sales were there in Windsor, CT on December 4, 2011?
2. How many pitches and sales were there in Albany, NY on December 4, 2011?
3. How many sales were there on December 3, 2011?
4. For the weekend of December 3 and December 4, 2011, what location had the most sales?
5. Though not shown in the sample data, what are the monthly sales totals for Windsor, CT for the last three months (i.e., the last quarter of 2011)?
6. What are the monthly sales totals for all locations for the last three months?
7. What are the least productive locations (i.e., which locations have the least sales)?

Submission Instructions

To submit your work, create a single ZIP file (or compressed folder) containing all of your source files. Use your Saint Rose ID (e.g., `goldschmidtd168`) as the name of the ZIP file (i.e., `goldschmidtd168.zip`).

Though entirely optional, you can include a simple `README.txt` file with notes or instructions.

Email your ZIP file to `goldschmidt@gmail.com` (with a subject line of "`CSC 380/530 Project 3`").