CSc 484
Database Management Systems

Ken Gamradt

Spring 2024

Introduction to SQL (I)

Overview of the SQL Language

- SQL: **Structured Query Language**
  - Most widely used database query language
- Originally developed by IBM in early 1970s
- Many products now support the SQL language
- SQL has clearly established itself as the standard relational database language

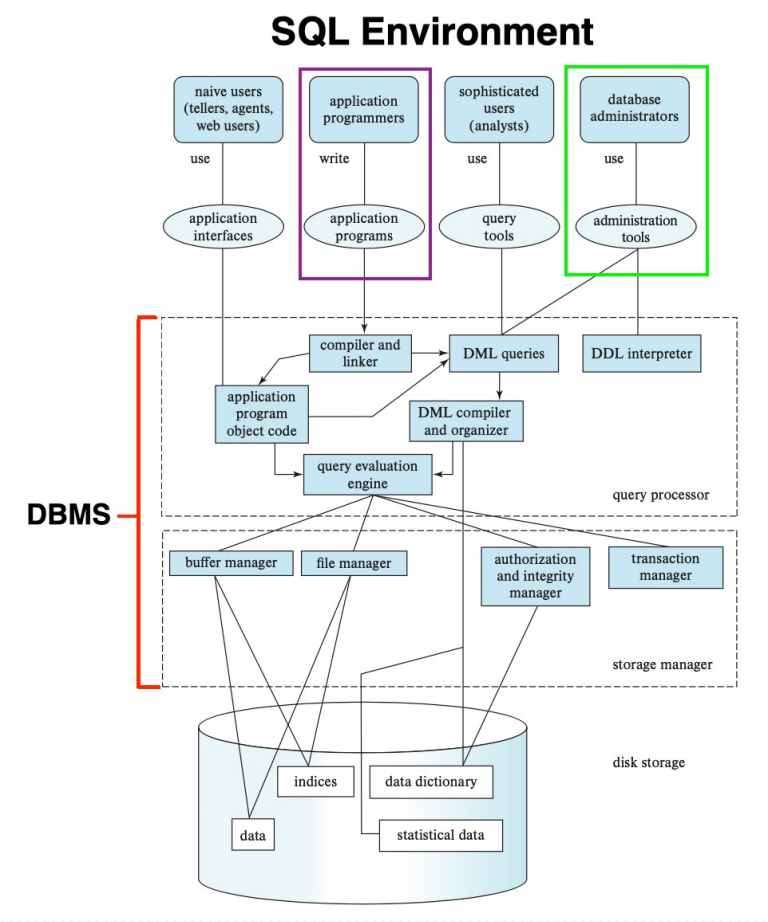
SQL

- Consists of several parts:
  - **Data-Definition language (DDL)**
    - Provides commands for defining relation schemas, deleting relations, and modifying relation schemas.
  - **Data-Manipulation language (DML)**
    - Provides the ability to query information from the database and to insert tuples into, delete tuples from, and modify tuples in the database
  - **Integrity**
    - DDL includes commands for specifying integrity constraints that the data stored in the database must satisfy.
    - Updates that violate integrity constraints are not allowed
  - **View definition**
    - The SQL DDL includes commands for defining views

SQL

- Consists of several parts:
  - **Transaction control**
    - SQL includes commands for specifying the beginning and ending points of transactions.
  - **Embedded and dynamic SQL**
    - Define how SQL statements can be embedded within general-purpose programming language
  - **Authorization**
    - The SQL DDL includes commands for specifying access rights to relations and views

SQL Environment



Basic Types for Attributes

- The SQL standard supports a variety of built-in types:
  - **char(n)**: fixed-length character string with user-specified length n
    - **character**: can be used instead
  - **varchar(n)**: variable-length character string with user-specified maximum length n
    - **character varying** can be used instead
  - **int**: integer
  - **smallint**: small integer – machine-dependent
  - **numeric(p, d)**: fixed-point number with user-specified precision
    - p digits (plus sign), and d of p digits are right of the decimal point
    - exact value stored
  - **real, double precision**: floating-point and double-precision floating-point numbers with machine-dependent precision
  - **float(n)**: floating-point number with precision of at least n digits
- Each type may include a special value: **null**

Basic Types for Attributes

- char(n) VS varchar(n)

value	char(4)	varchar(4)
‘’	‘ _ _ _ _’	‘’
‘ab’	‘ab _ _’	‘ab’
‘abcd’	‘abcd’	‘abcd’

- It is recommended to always use the varchar instead of char to avoid these problems

Basic Types for Attributes

- For the constants in SQL statements
  - All non-numeric data values must be enclosed in single quotes
  - All numeric data values must not be enclosed in single quotes

Writing SQL Commands

- An SQL statement consists of **reserved words** and **user-defined words**
- **Reserved words**:
  - Fixed part of the SQL language and have a fixed meaning
  - must be spelled exactly as required and cannot be split across lines
- **User-defined words**:
  - Made up by the user and represent the name of various database objects such as tables, columns, views, and so on
- Semicolon (;) is optionally used to terminate a statement

Writing SQL Commands

- Most components of an SQL statement are case-insensitive
  - the letters can be typed in either upper- or lowercase
  - Such as reserved word, relation names and attribute names

- Follow the extended form of the **Backus-Naur Form** (BNF) notation to define SQL statements
  - Uppercase letters are used to represent reserved words

Writing SQL Commands

- Although SQL is free-format, a statement is more readable if indentation and lineation are used:
  - Each clause in a statement should begin on a new line
  - The beginning of each clause should line up with the beginning of other clauses
  - If a clause has several parts, they should each appear on a separate line and be indented under the start of the clause to show the relationship

Writing SQL Commands

- The general form of the **create table** command is:

create table *r*
$(A_1 \quad D_1,$
$A_2 \quad D_2,$
$\ldots,$
$A_n \quad D_n,$
$\langle \text{integrity-constraint}_1 \rangle,$
$\ldots,$
$\langle \text{integrity-constraint}_k \rangle);$

```
create table department(
    dept_name    varchar(20),
    building     varchar(15),
    budget       numeric(12, 2),
    primary key (dept_name)
);
```

- Where r is the name of the relation
- Each A_i is the name of an attribute in the schema of relation r
- Each D_i is the domain of attribute A_i
  - D_i specifies the type of attribute A_i along with optional constraints that restrict the set of allowed values for A_i

Basic Schema Definition – Define a Relation

- SQL support a number of different integrity constraints:
  - **PRIMARY KEY** (Aj_1, Aj_2, … , Aj_n)
    - The attributes Aj_1, Aj_2, … , Aj_n form the primary key for the relation
    - The primary key attributes are required to be nonnull and unique
  - **FOREIGN KEY** (Ak_1, Ak_2, … , Ak_n) references S
    - The values of attributes (Ak_1, Ak_2, … , Ak_n) for any tuple in the relation must correspond to values of the primary key attributes of some tuple in relation S
  - **NOT NULL**: the not null constraint specifies the null value is not allowed for that attribute
  - …

Basic Schema Definition – Define a Relation

```
create table instructor(
    ID          varchar(5),
    name        varchar(20) not null,
    dept_name   varchar(20),
    salary      numeric(8, 2),
    primary key (ID),
    foreign key (dept_name) references department
);
```

- The referenced attribute list can be eliminated if the name are the same as in the referencing relation
  - ***foreign key (dept_name) references(dept_name));***

Basic Schema Definition – Remove a Relation

- To remove a relation from an SQL database, we use the **drop table** command
- The drop table command deletes all information about the dropped relation from the database
- The command:

drop table r;

Basic Schema Definition – Alter a Table

- We use the **alter table** command to add attributes to an existing relation
- All tuples in the relation are assigned null as the value for the new attribute
- The form of the alter table command is:

alter table *r* ***add*** *A D;*

R: the existing relation
A: the name of the attribute to be added
D: the type of added attribute

- Drop attributes from a relation by the command

alter table *r* ***drop*** *A;*

R: the existing relation
A: the name of the attribute to be dropped

Insertion

- Insert data into a relation
  - Insert one tuple into the database

The order has to match the corresponding attributes listed in the relation schema

```
insert into course
    values('CSC-484', 'Database Management', 'EECS', 3);
```

  - Or

The order doesn't have to match the attributes order in the relation schema. But has to match the values

```
insert into course (course_id, title, dept-name, credit)
    values('CSC-484', 'Database Management', 'EECS', 3);
```

Identical

```
insert into course (title, course_id, dept-name, credit)
    values('Database Management', 'CSC-484', 'EECS', 3);
```

Insertion

- Insert data into a relation
  - Insert one tuple into the database

    insert into *course*

    values*('CSC-484', 'Database Management', 'EECS', 3),*

    ('CSC-300', 'Data Structures', 'EECS', 3);
  - Insert tuples with values only on some attributes of the schema
    - remaining attributes are assigned a null value

    insert into *course (course_id, title, dept-name, credit)*

    values*('CSC-484', 'Database Management', 'EECS');*

Identical  **Some attributes with values**

insert into *course*

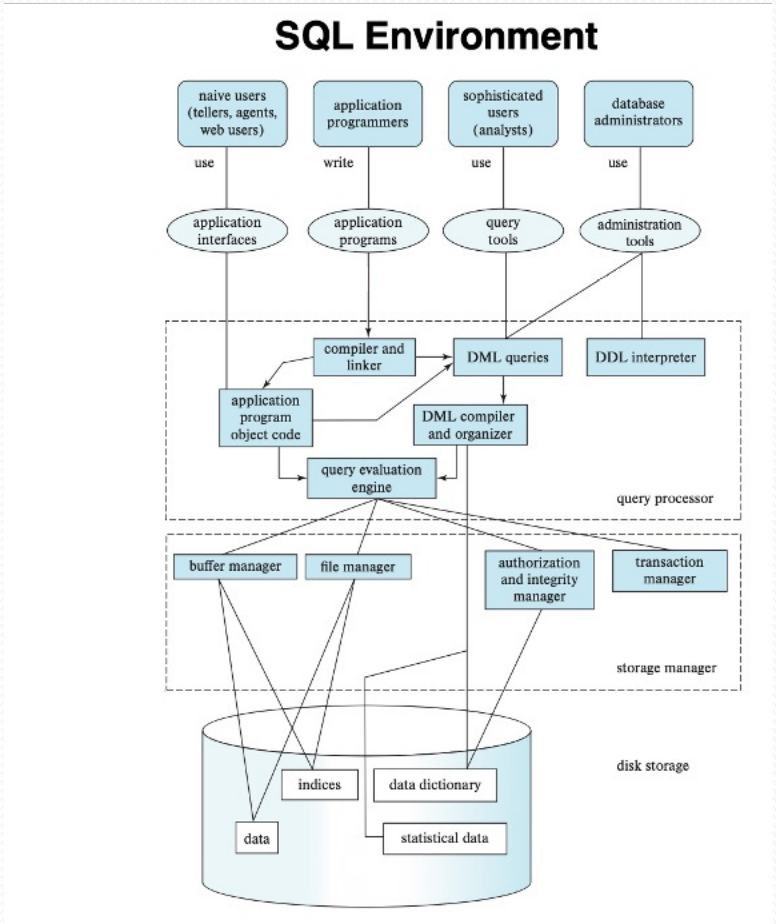
values*('CSC-484', 'Database Management', 'EECS', null);*

Clarify the null attribute values

Tools

- Popular relational database systems:
  - IBM DB2, Microsoft SQL Server, Oracle
  - MariaDB, MySQL, PostgreSQL
- Most database systems provide a command line interface for submitting SQL commands
- Most database systems also provide graphical user interfaces (GUIs) which:
  - Simplify browsing the database
  - Simplify creating and submitting queries
  - Simplify administering the database
  - MySQL + HeidiSQL / pnpMyAdmin
  - Microsoft SQL Server + SQL Server Management Studio / Azure Studio
  - Oracle + Oracle SQL Developer
  - PostgreSQL + pgAdmin

Tools


SQL Environment
naive users (tellers, agents, web users)
application programmers
sophisticated users (analysts)
database administrators
use
write
use
use
application interfaces
application programs
query tools
administration tools
compiler and linker
DML queries
DDL interpreter
application program object code
DML compiler and organizer
query evaluation engine
query processor
buffer manager
file manager
authorization and integrity manager
transaction manager
storage manager
disk storage
indices
data dictionary
data
statistical data

Acknowledgements

- WIKIPEDIA
  - https://en.wikipedia.org/wiki/Backus-Naur_form