DB笔记

Principles of Database

数据库原理

1. Overview

In computing, a database is an organized collection of data stored and accessed electronically. A database management system (DBMS) is the software that interacts with end users, applications, and the database itself to capture and analyze the data.
DBMS is software system that enables users to define, create, maintain and control access to the database.
Examples of DBMS's include MySQL, PostgreSQL, Microsoft SQL Server, Oracle Database, and Microsoft Access.

2. Data Model

Abstraction process of objective objects—two-step abstraction:

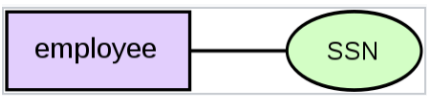
1. The objective objects in the real world are abstracted as conceptual models;
2. Convert a conceptual model to a data model supported by a DBMS.

2.1 Conceptual Data Model

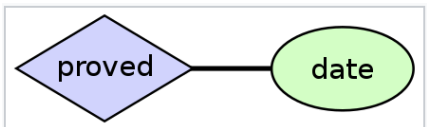
A conceptual data model is a map of concepts and their relationships used for databases. Specifically, it describes the things of significance to an organization (entity classes), and its characteristics (attributes) and the associations between pairs of those things of significance (relationships).

- Entity-Relationship Model
  The entity-relationship model (or ER model) abstracts the real world into entities and their relationships. It mainly includes three parts: entity (classes), attribute and relationships.

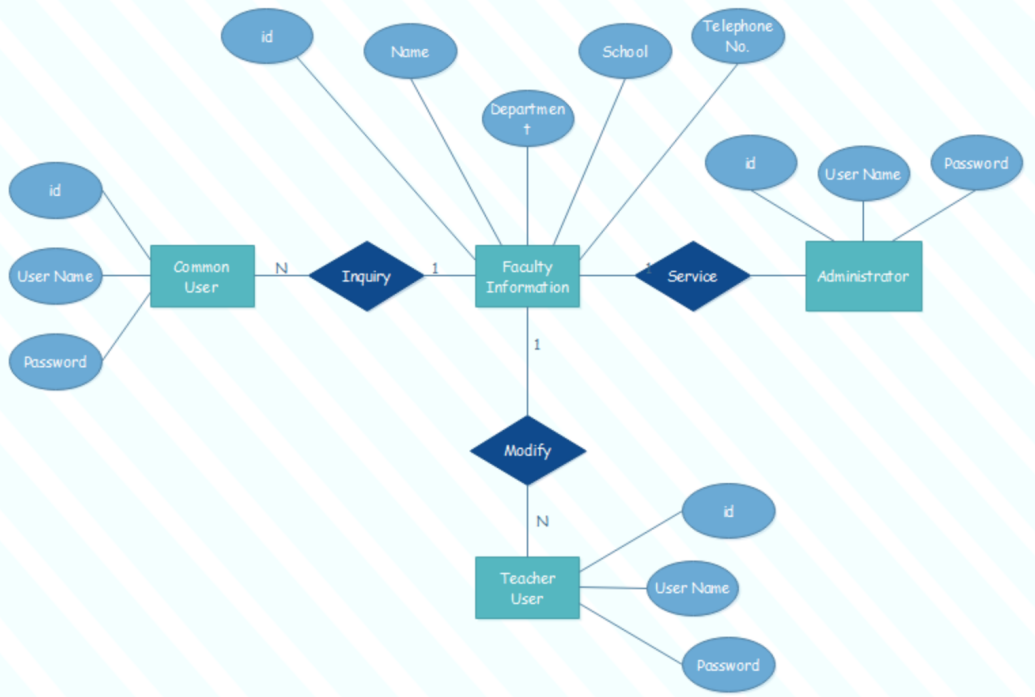
Two related entities



An entity with an attribute



A relationship with an attribute



2.2 Database Model

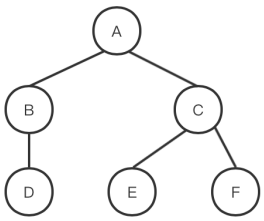
- Hierarchical model

  A hierarchical database model is a data model in which the data are organized into a tree-like structure. The connection between many entities in the real world itself presents a natural hierarchical relationship, such as administrative agencies, family relationships, and so on.

  The data are stored as records which are connected to one another through links. A record is a collection of fields, with each field containing only one value. The hierarchical database model mandates that each child record has only one parent, whereas each parent record can have one or more child records.

  Examples of hierarchical data represented as relational tables

ID	Value	Parent ID
A	1.1	NULL
B	2.1	A
C	2.2	A
D	3.1	B
E	3.2	C
F	3.2	C



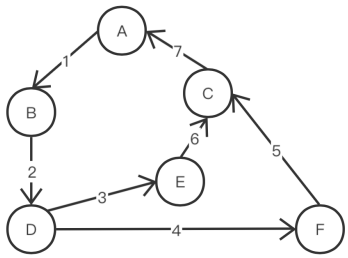
- Network model

  The network model is a database model conceived as a flexible way of representing objects and their relationships. Its distinguishing feature is that the schema, viewed as a graph in which object types are nodes and relationship types are arcs.

  Examples of network data represented as relational tables

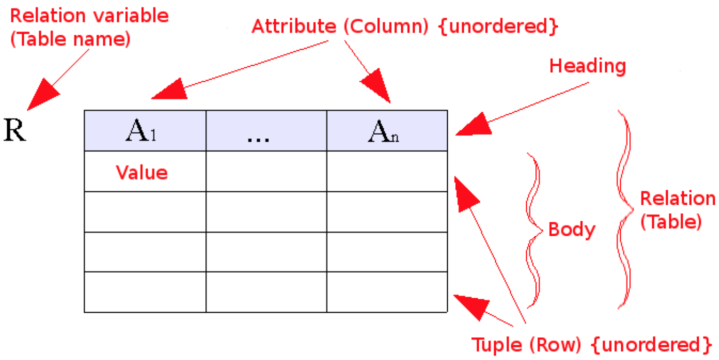
Entity	
EID	Value
A	10001
B	10002
C	10003
D	10004
E	10005
F	10006

Relationship			
RID	Name	Previous ID	Next ID
1	XX	A	B
2	XXX	B	D
3	X	D	E
4	XXXX	D	F
5	XX	F	C
6	XXX	E	C
7	XXX	C	A



- Relational model

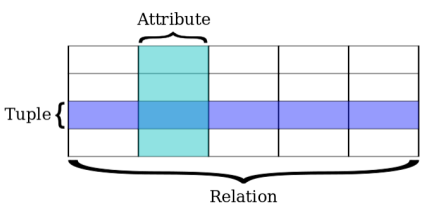
  This model organizes data into one or more tables (or "relations") of columns and rows, with a unique key identifying each row. Rows are also called records or tuples. Columns are also called attributes. Generally, each table/relation represents one "entity type". The rows represent instances of that type of entity and the columns representing values attributed to that instance.



  It is the most commonly used data model at present.

3. Relational Database

- Relational data structure: Relational data structures use two-dimensional tables to organize data, and two-dimensional tables are called relationships in relational databases. A relational database is a collection of tables, and a relational database makes users feel that the data is just a table. Here the table is a logical structure rather than a physical structure, how the physical layer is stored is invisible to the user.
- Terminology



  - relation, tuple, attribute: A relation is defined as a set of tuples that have the same attributes.

    - domain: A domain describes the set of possible values for a given attribute, and can be considered a constraint on the value of the attribute
    - primaiy key: Each relation/table has a primary key. primary key uniquely specifies a tuple within a table
    - foreign key: A foreign key is a field in a relational table that matches the primary key column of another table.
- Relational operations: Union, Intersection, Select, Join, Query, Insert, Delete, Update. The operation object of the relational model is one or more tables.
- Constraints:
    - domain constraints: Limit the value range of attributes in the relation to prevent conflict between attribute values and application semantics.
    - entity integrity: There must be a primary key in each relation, so that the primary key is uniquely identified as a tuple. Primary key cannot be empty.
    - referential integrity: Referential integrity is a property of data stating that all its references are valid. In the context of relational databases, it requires that if a value of one attribute (column) of a relation references a value of another attribute (either in the same or a different relation), then the referenced value must exist.

4. SQL (of MySQL)

SQL (Structured Query Language) is the general language and standard language for users to operate relational databases. The scope of SQL includes data query, data manipulation (insert, update and delete), data definition (schema creation and modification), and so on.

SQL is case insensitive by default.

4.1 Database operations

```
-- create database
create DATABASE db_name;
-- delete database
drop DATABASE db_name;
-- select database
use db_name;
-- show all of database
select database();
```

4.2 Table operations

4.2.1 Create Table

Format:

```
create TABLE table_name (
  column_name1 data_type [integrity_constraint_definition],
  column_name2 data_type [integrity_constraint_definition],
```

```
...);
```

Example:

```
create TABLE interviewer (
  IID int NOT NULL, /* NOT NULL: not empty */
  Name varchar(20) NOT NULL,
  Sex enum('F','M'),
  PRIMARY KEY (IID), /* PRIMARY KEY: set primary key */
);
create TABLE student (
  SID int NOT NULL,
  IID int,
  Name varchar(20) NOT NULL,
  IdNumber varchar(30) NOT NULL UNIQUE,  /* UNIQUE: value unique */
  AuditTimes int DEFAULT 0, /* DEFAULT: set default value */
  CHECK (AuditTimes>=0) /* CHECK: set range of attribute */
  PRIMARY KEY (SID),
  /* FOREIGN KEY: set foreign key and its original table */
  FOREIGN KEY (P_Id) REFERENCES interviewer(IID),
);
```

4.2.2 Delete Table

```
drop TABLE table_name;
```

4.2.3 Update Table

```
-- add column
alter TABLE table_name
add column_name datatype;
-- delete column
alter TABLE table_name
drop COLUMN column_name;
-- alter column
alter TABLE table_name
modify COLUMN column_name datatype;
-- alter column name
alter TABLE table_name
change COLUMN column_name column_name2 datatype;
-- show column
show COLUMNS FROM table_name;
```

4.2.3 Show all of Table

```
show tables;
```

4.3 Data operations

Insert / Update / Delete

```
-- insert data
```

```
INSERT INTO interviewer
(FirstName, LastName, IDNumber, Sex, Age)
VALUES
('Yang', 'Yue', '123X123', 'F', 25);
-- update data
update student set AuditStatus = 2
where SID = 1;
   -- ---------------------- --
update student set AuditStatus = 4
where exists
  (select * from result
  where SID = student.SID and Passed is null);
-- delete data
delete from result
where SID = 2 and Passed is null;
```

Select

```
-- select all
select * from table_name;
-- select with WHERE
select concat(FirstName, LastName) Name, Result
from student
where Result IS NOT NULL AND Age BETWEEN 20 AND 30;
-- select with WHERE and IN
select concat(FirstName, LastName) Name, Result
from student
where Passed IN ('Y', 'N');
-- select with WHERE and LIKE
select concat(FirstName,LastName) Name, APSPassed, IDNumber
from student
where IDNumber like '%1988%';
-- order by
select concat(FirstName,LastName) Name, Sex, Age
from student
order by Age [DESC]; /*DESC means sorting in descending order*/
-- group by
select SID as student id, count(*) as times
from result
group by SID;
```

Operation

```
select count(*) from student; /* count(* or column_name) */
select sum(age) from student;
select avg(age) from student;
select max(age) from student;
select min(age) from student;
```

Multi-table join

```
-- join/ inner join
select student.SID, Name, Sex, Age, NumOfInterv, Passed
from student
join result
on student.SID = result.SID
```

4.4 View operations

```
-- create view
create view vStudent
as select SID as ID, concat(FirstName,LastName) as Name, Sex, Age
from student;
-- drop view
drop view vStudent;
```

5. Database Normalization

Database normalization is the process of structuring a database, usually a relational database, in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity.

5.1 1NF

To satisfy First normal form, each column of a table must have a single value. Columns which contain sets of values or nested records are not allowed.

5.2 2NF

To conform to 2NF and remove duplicities, every non candidate-key attribute must depend on the whole candidate key, not just part of it.

5.3 3NF

In 3NF, every non candidate-key attribute mustn't has any transitive functional dependency on the primary key. Usually in database design, it is generally required to achieve 3NF.

References:

思维导图

Conceptual Schema - Wikipedia

Data Model - Wikipedia

Referential integrity - Wikipedia

Database Normalization - Wikipedia

SQL - Wikipedia

SQL教程 - W3school

SQL教程 - 菜鸟RUNOOB

MySQL教程 - 菜鸟RUNOOB