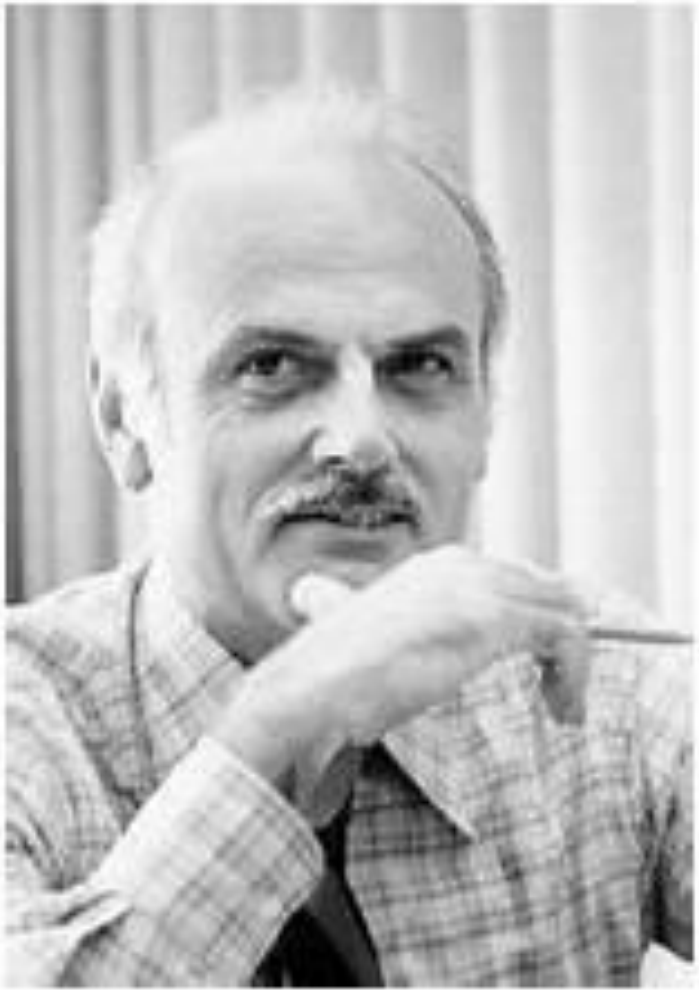
Principles of Database Systems

Introduction to SQL(1)

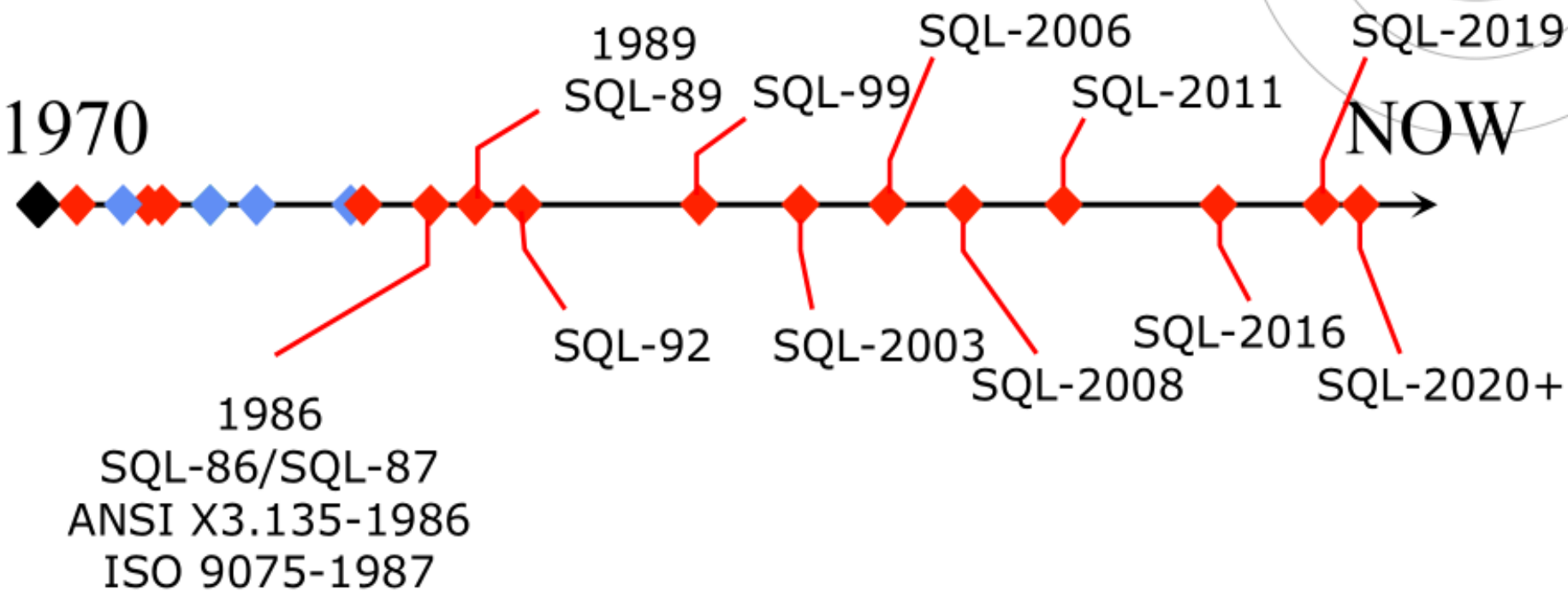
Edgar Frank Codd



埃德加·弗兰克·科德（Edgar Frank Codd，1923－2003）是密执安大学哲学博士，IBM公司研究员，被誉为“关系数据库之父”，并因为在数据库管理系统的理论和实践方面的杰出贡献于1981年获图灵奖。1970年，科德发表题为"**A Relational Model of Data for Large Shared Data Banks**"（大型共享数据库的关系数据模型）的论文，文中首次提出了数据库的关系模型。由于关系模型简单明了、具有坚实的数学理论基础，所以一经推出就受到了学术界和产业界的高度重视和广泛响应，并很快成为数据库市场的主流。20世纪80年代以来，计算机厂商推出的数据库管理系统几乎都支持关系模型，数据库领域当前的研究工作大都以关系模型为基础。

Overview of the SQL Query Language

History of SQL



History of SQL

- IBM Sequel language developed as part of **System R** project at the IBM San Jose Research Laboratory

- Renamed Structured Query Language (SQL,结构化查询语言)

- ANSI and ISO standard SQL:
  - SQL-86, SQL-89, SQL-92
  - SQL:1999, SQL:2003, SQL:2008

  - ANSI: the American National Standards Institute
  - ISO: the International Organization for Standardization

Database Languages

- Database Languages as human-machine interfaces
  - **Data-Manipulation Language, DML**
    (数据操纵语言)
  - **Data-Definition Language, DDL**
    (数据定义语言)

姓名	生日	身高	项目	时间	国家
博尔特	1986.821	196	100米跑	9’79	牙买加
苏炳添	1989.8.29	172	100米跑	~~9’99~~ 9’83	中国
宁泽涛	1993.3.6	191	100米自	47’65	中国
菲尔普斯	1985.6.30	193	100米蝶	50’58	美国

Database Languages

- **Data Manipulation Language (DML)**
  - Language for accessing and manipulating the data organized by the appropriate data model
  - DML also known as query language

- Two classes of languages
  - **Procedural** (过程化DML)– user specifies what data is required and how to get those data
  - **Declarative (nonprocedural)** (声明式DML) – user specifies what data is required without specifying how to get those data

- Query (查询):a statement requesting the retrieval of information

- SQL is the most widely used query language

Constituent Parts of SQL (SQL组成部分)

- The SQL language has several parts:
  - Data-definition language (DDL)
  - Data-manipulation language (DML)
  - Integrity (完整性) (included in DDL)
  - View definition (视图定义) (included in DDL)
  - Transaction control (事务控制)
  - Embedded SQL and dynamic SQL (嵌入式SQL及动态SQL)
  - Authorization (授权)

Implementation of SQL (SQL实现)

- Commercial systems offer most, if not all, SQL-92 features, plus varying feature sets from later standards and special proprietary features.

- Not all examples here may work on your particular system.
  - E.g. the “natural join”(自然连接) is not implemented in Microsoft SQL Server.

SQL Data Definition

Functions of DDL

- The SQL DDL allows specification of not only a set of relations, but also information about each relation, including:
  - **The schema for each relation.**
  - **The types of values associated with each attribute.**
  - **The integrity constraints.**
  - The set of indices to be maintained for each relation.
  - The security and authorization information for each relation.
  - The physical storage structure of each relation on disk.

Basic Types

- **char(n)**. Fixed length character string, with user-specified length n.
- **varchar(n)**. Variable length character strings, with user-specified maximum length n.
- **int**. Integer (a finite subset of the integers that is machine-dependent).
- **smallint**. Small integer (a machine-dependent subset of the integer domain type).
- **numeric(p,d)**. Fixed point number, with user-specified precision of p digits, with d digits to the right of decimal point.
- **real**, **double precision**. Floating point and double-precision floating point numbers, with machine-dependent precision.
- **float(n)**. Floating point number, with user-specified precision of at least n digits.
- **date, time, timestamp, interval**
- Each type may include a special value called the **null** value

データ型（例）

- 文字列
  - character(長さ), char(長さ) : 固定長文字列
  - character varying(長さ), varchar(長さ) : 可変長文字列
- 数値
  - integer, int : 整数
  - numeric(精度,位どり): 固定小数点(小数点以下桁数が指定可能)
    - numeric(5,2)は全体の桁数が5ケタで、小数点以下は2桁
- 実数
  - float [(精度)] : 浮動小数点数値データ
  - real : 4バイトで表現できる実数(精度は8桁)
  - double precision : 8バイトで表現できる実数(精度は15桁)
- 日付
  - date : 年月日
  - time : 時刻
  - timestamp : 日付と時刻

Basic Schema Definition-Create

- We define an SQL relation by using the **create table** command.

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

Basic Schema Definition-Create

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

  create table r
  (A_1 D_1,
  A_2 D_2,
  ...,
  A_n D_n,
  (integrity-constraint$_1$),...,
  (integrity-constraint$_k$));

  - r is the name of the relation
  - each A_i is an attribute name in the schema of relation r
  - D_i is the data type of values in the domain of attribute A_i

Integrity Constraints in DDL

- SQL supports a number of different integrity constraints:
  - **primary key** (A_{j1}, A_{j2},..., A_{jm}): The primary-key specification says that attributes A_{j1}, A_{j2},..., A_{jm} form the primary key for the relation. The primary key attributes are required to be *nonnull* and *unique*;
  - **foreign key** **(A_{k1}, A_{k2},..., A_{kn}) references s**:The foreign key specification says that the values of attributes (A_{k1}, A_{k2},..., A_{kn}) for any tuple in the relation must correspond to values of the primary key attributes of some tuple in relation *s*.

Integrity Constraints in DDL (Cont.)

- SQL supports a number of different integrity constraints:
  - **not null**: The not null constraint on an attribute specifies that the null value is not allowed for that attribute
  - Declare name and budget to be **not null**
  - name **varchar**(20) **not null**
  - budget **numeric**(12,2) **not null**

Integrity Constraints in DDL (Cont.)

- **unique** (A_1, A_2, …, A_m)
  - The unique specification states that the attributes A_1, A_2, … A_m form a **candidate key**.
  - Candidate keys are **permitted to be null** (in contrast to primary keys).

Integrity Constraints in DDL (Cont.)

- The **check** clause is applied to relation declaration
  - check (P), where P is a predicate which must be satisfied by every tuple in the relation.

- Example: ensure that the budget of a department must be greater than $0.00
  - **create table** department
    (dept name **varchar (20),**
    building **varchar (15),**
    budget **numeric (12,2),**
    primary key (dept name**)**
    check(budget>0**));**

Integrity Constraints in DDL (Cont.)

- **SQL prevents any update to the database that violates an integrity constraint.**
  - For example, a tuple has null value for any primary key attribute.

Basic Schema Definition-Drop

- The **drop table** command **deletes all information (tuples and schema)** about the dropped relation from the database

drop table *r*

Basic Schema Definition-Alter

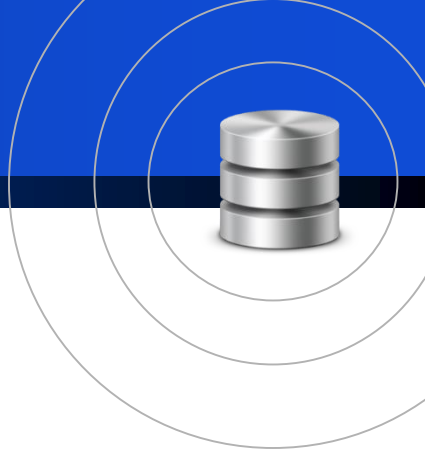
- The **alter table** command is used to add or delete/drop attributes to an existing relation

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

  where *A* is the name of the attribute to be added to relation *r* and *D* is the type of *A*
  - all tuples in the relation are assigned null as the value for the new attribute

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

  where *A* is the name of an attribute of relation *r*
  - dropping of attributes not supported by many databases



SQL Data Manipulation

Data Manipulation

- A newly created table is empty initially, we can use **insert** command to load data into the table

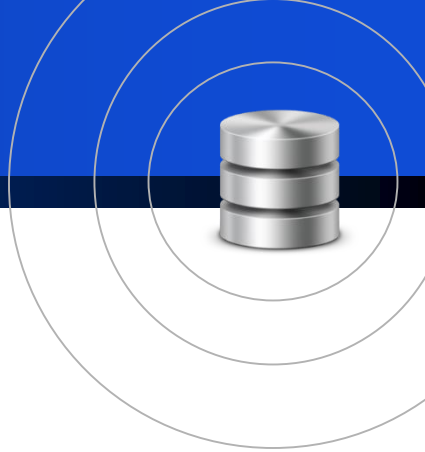
  insert into *instructor*
  values (10211, 'Smith', 'Biology', 66000);

- The **delete** command removes tuples from the table

  delete from account

- The **update** command changes a value in a tuple without changing all values in the tuple.

  update instructor
  set salary= salary * 1.05;



SQL Data Query

SQL Data Query

- The SQL data-manipulation language (DML) provides the ability to query information, and insert, delete and update tuples

- "Query" (查询) could be generalized definition
  - Define (定义) , **retrieve (检索)**, modify (修改), control (控制) etc. on DB
  - Define→DDL(create, alter, drop)
  - Modify→DML(insert, update, delete)
  - **Retrieve→DML(select)**
  - Control→DCL(grant,revoke)

Key points

- Always keep in mind:
  - SQL statements are case insensitive (大小写不敏感)
    - E.g. Name ≡ NAME ≡ name
  - SQL statements must follow the fixed syntax (固定语法)
  - **Both the input and output of SQL statements are table(loosely speaking, relation)**, because SQL statements are the implementations of relational operations (关系运算). This point is especially important for understanding the underlying principle of query, that is SELECT statement.

Database Explained

院系（department）

招收学生

设立课程

聘任老师

开教学班

指导:
advisor

课程（course）

导师（instructor）

学生（student）

教学: teach

选取: take

教室（classroom）

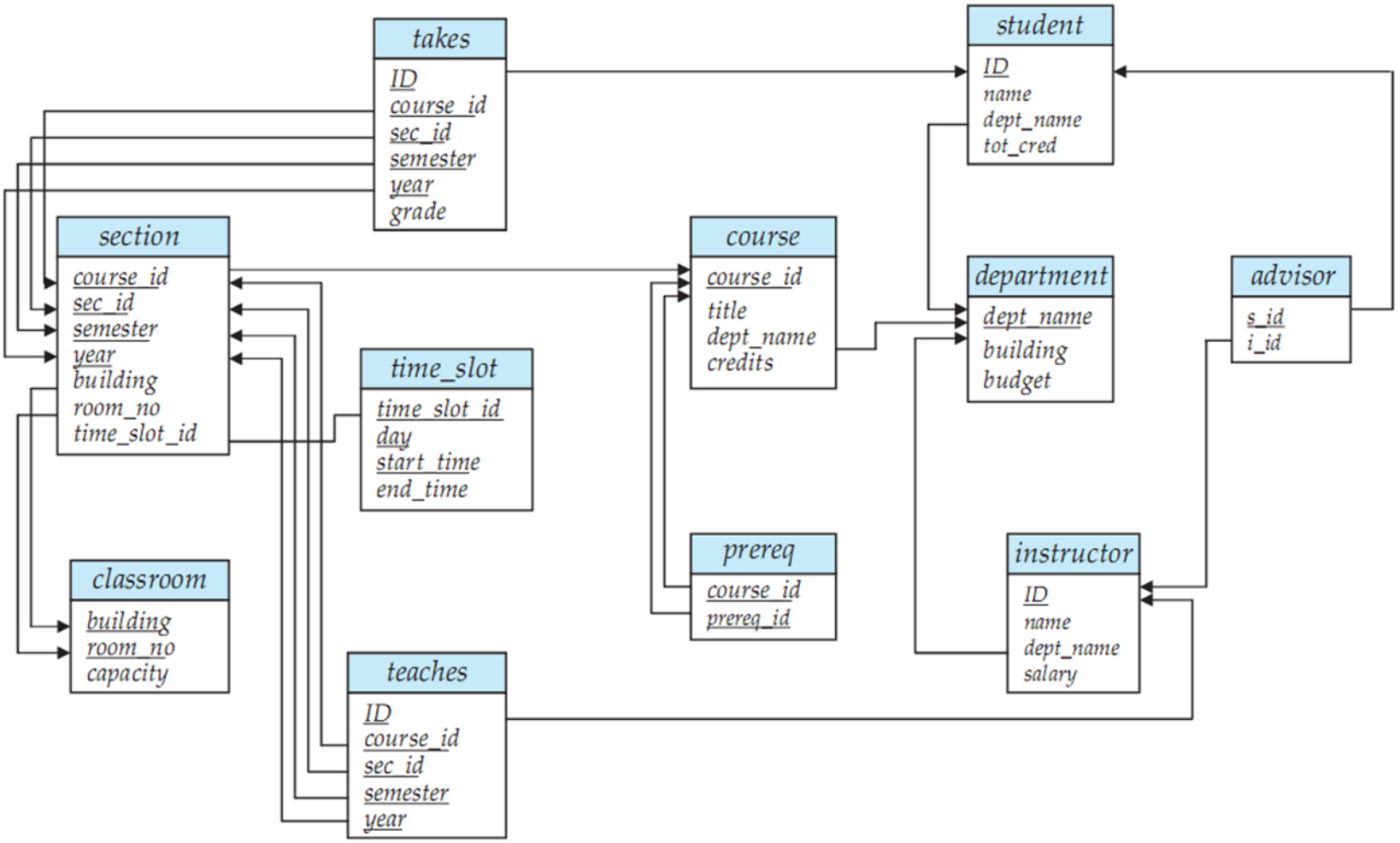
开课（section）

使用教室

上课时间

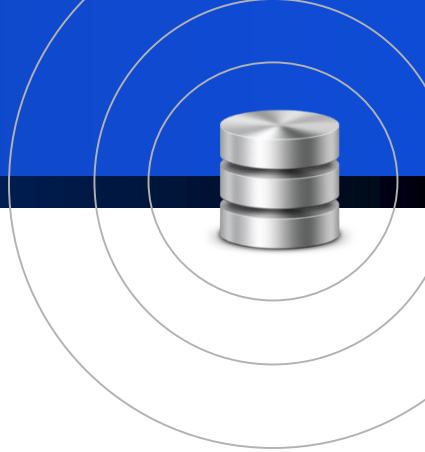
时间段（time_slot）

Database Used

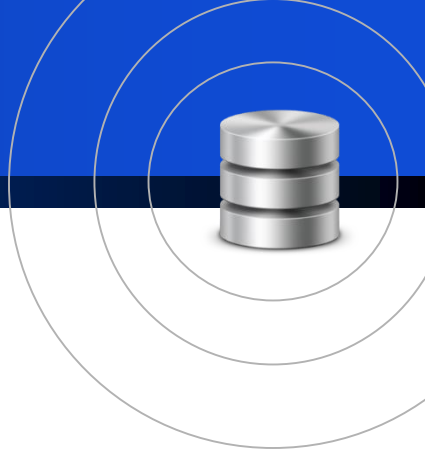


Database Used

classroom(*building*, *room_number*, *capacity*)
department(*dept_name*, *building*, *budget*)
course(*course_id*, *title*, *dept_name*, *credits*)
instructor(*ID*, *name*, *dept_name*, *salary*)
section(*course_id*, *sec_id*, *semester*, *year*, *building*, *room_number*, *time_slot_id*)
teaches(*ID*, *course_id*, *sec_id*, *semester*, *year*)
student(*ID*, *name*, *dept_name*, *tot_cred*)
takes(*ID*, *course_id*, *sec_id*, *semester*, *year*, *grade*)
advisor(*s_ID*, *i_ID*)
time_slot(*time_slot_id*, *day*, *start_time*, *end_time*)
prereq(*course_id*, *prereq_id*)



```
create table instructor
        (ID                         varchar(5),
         name                       varchar(20) not null,
         dept_name                  varchar(20),
         salary                     numeric(8,2) check (salary > 29000),
         primary key (ID),
         foreign key (dept_name) references department
          );
```



Thanks