اصول طراحی پایگاه داده ها

اسماعیل نورانی

فصل ۳

پایگاه داده رابطه ای

مطالبی که در این فصل خواهیم خواند

- معماری پایگاه داده
- کلید
- طراحی پایگاه داده رابطه ای
- تبدیل نمودار ER به جداول
- قواعد جامعیت

سطوح معماری بانک اطلاعاتی

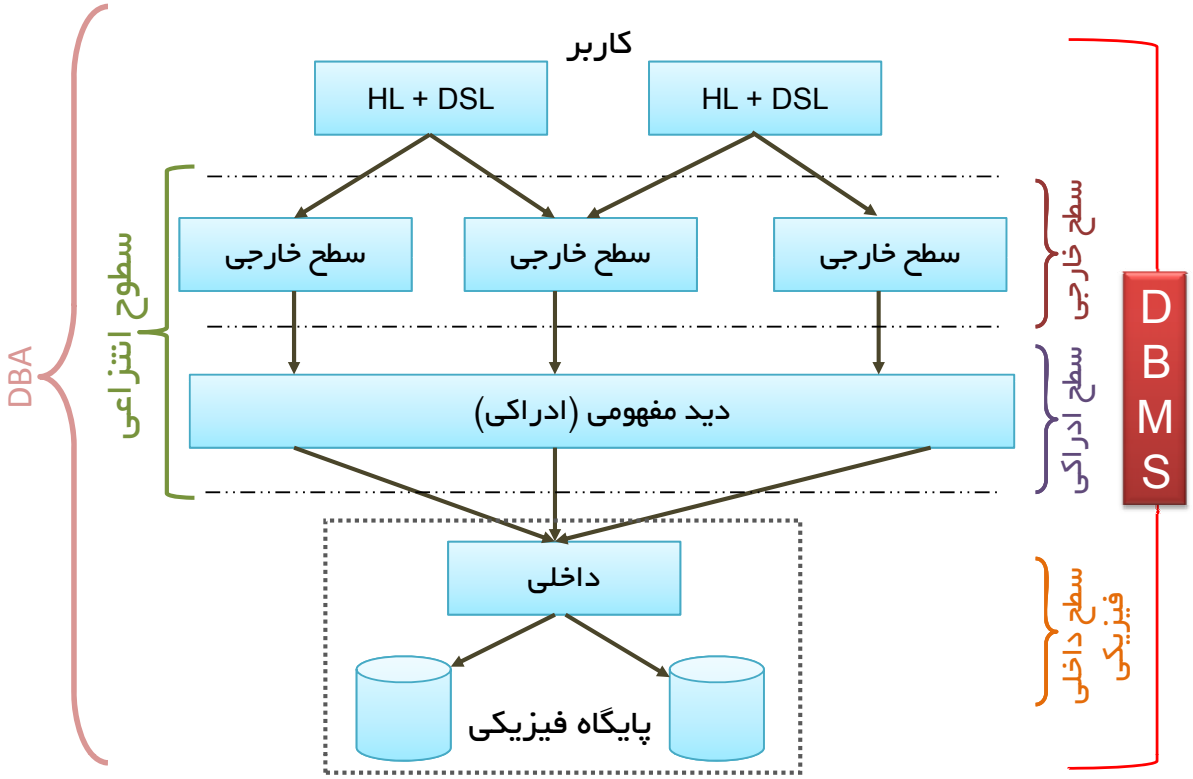
- یک سیستم پایگاه داده در حالت استاندارد از سه سطح تشکیل شده است.

دید اول ، سطح خارجی ، دید کاربر ، (External View)

دید دوم ، سطح ادراکی ، سطح مفهومی ، (Conceptual View)

دید سوم ، سطح داخلی ، سطح فیزیکی ، (Internal Level)

سطوح معماری بانک اطلاعاتی



سطوح معماری بانک اطلاعاتی

- دید اول ، سطح خارجی ،دید کاربر ، (External View)

این دید ، دید کاربران نهایی است که به کمک زبان های (Host language , HL) مانند زبان C# و (Data Sub Language , DSL) مانند زبان SQL ، تعریف شده است.

ممکن است دید کاربران مختلف نسبت به یک موضوع متفاوت باشد، به طور مثال در سیستم دانشگاه دیدی که کارمند آموزشی از سیستم دارد ، ممکن است با دید کارمند امور مالی متفاوت باشد.

سطوح معماری بانک اطلاعاتی

- دید دوم ، سطح ادراکی ، سطح مفهومی ، (Conceptual View)

این دید ، خاص طراح پایگاه داده می باشد.

در حقیقت مدلی مفهومی از پایگاه داده ارائه می شود ، مانند نمودار ER

دید طراح ، دید جامع همه دیدهای خارجی است.

- دید سوم ، سطح داخلی ، سطح فیزیکی ، (Internal Level)

این سطح حاوی فایل های فیزیکی پایگاه داده است که قرار نیست با جزئیات آن به طور مستقیم درگیر شویم و این وظیفه بر عهده DBMS است.

سطوح معماری بانک اطلاعاتی ـ نکات

- (Database Management System , DBMS) ناظر بر همه ی سطوح می باشد.
- به دو سطح خارجی و مفهومی ، سطوح انتزاعی (Abstractive) نیز گفته می شود.
- همه ی لایه ها و اعمال پایگاه داده ها توسط (Database Administrator , DBA) کنترل می شود.
- رییس DBA را DA (Data Administrator) گویند و ممکن است فردی فنی نباشد.

کلید

- کلید : اگر R مجموعه ای از صفات یک رابطه یا جدول باشد ، و K زیرمجموعه ای از این صفات باشد، می گوییم K کلید رابطه است ، اگر مقادیر K بین هیچ دو نمونه موجودیتی، تکراری نباشد؛ یعنی همه ی مقادیرش منحصر به فرد باشد. ($K \subseteq R$)
- انواع کلید :

ابر کلید (Super Key , SK)

کلید کاندید (Candidate Key , CK)

کلید اصلی (Primary Key , PK)

کلید خارجی (Foreign Key , FK)

کلید

- ابر کلید (Super Key , SK)

هر زیر مجموعه از صفات رابطه ای که خاصیت یکتا بودن را برقرار نماید.

مثال : ترکیب دوصفت (کد ملی ، نام خانوادگی) یک ابرکلید برای دانشجو است

- کلید کاندید (Candidate Key , CK)

ابرکلیدی که همه ی صفاتش برای برقراری خاصیت یکتا بودن ضروری می باشد؛ یعنی با حذف هر کدام، ممکن است خاصیت یکتا بودن از بین برود؛ در این صورت آن ابرکلید، کلید کاندید یا (Minimal) نیز می باشد.

مثال : شماره دانشجویی

شرایط کلید کاندید:

یکتا بودن و کاهش ناپذیر بودن. (هر کلید کاندید ، ابر کلید است ولی عکسش صادق نیست)

کلید

- کلید اصلی (Primary Key , PK)

هر کدام از کلید های کاندید را می توان بعنوان کلید اصلی انتخاب کرد.

معیارهای انتخاب کلید اصلی :

فیلدی که عددی است ، در اولویت قرار دارد

فیلدی که اهمیت بیشتری دارد ، مثلا برای دانشجو ، شماره دانشجویی بر شماره ملی آن دانشجو ارجحیت دارد.

کلید

- کلید خارجی (Foreign Key . FK)

صفتی از یک رابطه ، کلید خارجی محسوب می شود ، اگر در رابطه ی دیگر کلید اصلی باشد.

نحوه استفاده از کلید خارجی :

وسیله ای است برای پیوند دادن رابطه های بانک اطلاعاتی

طراحی پایگاه داده رابطه ای

- سه روش برای طراحی پایگاه داده رابطه ای (Relational Database , RDB) می باشد:

روش بالا به پایین (Top Down Design Method)

ابتدا مدل ER ایجاد شده ، سپس از روی آن ، رابطه ها و جداول ایجاد می گردد.

روش سنتز رابطه ای (Relational Synthesis)

همه ی صفات محیط ، جمع آوری شده و سپس با توجه به اصولی ، آنها را گروه بندی می کنیم ؛ این تجزیه تا جایی ادامه می یابد تا به حد ایده آل و مناسب برسیم.

روش ترکیبی (Hybrid method)

ابتدا از روش طراحی بالا به پایین شروع کرده ، سپس رابطه های را بدست آمده را تجزیه و نرمالتر می کنیم.

مثالهایی از جداول سیستم بانک

Account

account_number	*branch_name*	*balance*
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Customer

customer_name	*customer_street*	*customer_city*
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Green	Walnut	Stamford
Hayes	Main	Harrison
Johnson	Alma	Palo Alto
Jones	Main	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Putnam	Stamford
Williams	Nassau	Princeton

Loan

loan_number	*branch_name*	*amount*	*customer_name*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-155	*null*	*null*	Hayes

طراحی پایگاه داده رابطه ای

- یک طراحی خوب می بایست شرایط ذیل را دارا باشد:

۱. نمایش واضحی از محیط ارائه نماید.

۲. نمایش جامعی از همه ی نیازهای کاربران ارائه دهد.

۳. کمترین میزان افزونگی را داشته باشد.

۴. کمترین دشواری در انجام عمل ذخیره سازی را داشته باشد.

۵. کمترین میزان Null در آن ظاهر شود.

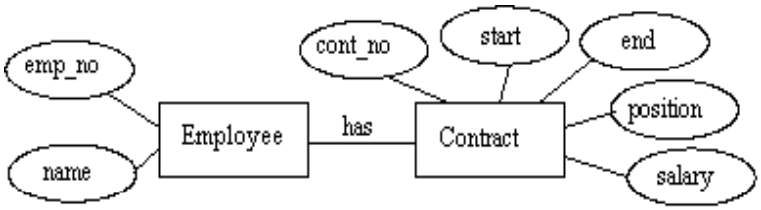
تبدیل نمودار ER به جداول

- حالت اول ، 1 : 1
- حالت دوم ، 1 : N
- حالت سوم ، N : M
- حالت چهارم ، پیاده سازی موجودیت ضعیف
- حالت پنجم ، پیاده سازی صفت چند مقداری
- حالت ششم ، پیاده سازی ارتباط IS - A
- حالت هفتم ، پیاده سازی ارتباط IS - A Part Of

تبدیل نمودار ER به جداول

- حالت اول ، 1 : 1

هر کارمند یک قرارداد دارد و هر قرارداد مربوط به یک کارمند است



Employee(emp_no, name, cont_no, start, end, position, salary)

PK

یا می تواند به صورت دو رابطه زیر تبدیل شود:

Employee (emp_no, name, contract_no)

PK *FK*

Contract(cont_no, start, end, position, salary)

PK

نکته :

اگر شرکت در ارتباط هر دو موجودیت الزامی باشد فقط یک رابطه کافی است

تبدیل نمودار ER به جداول

- حالت دوم ، 1 : N

Department(Dcode,title)

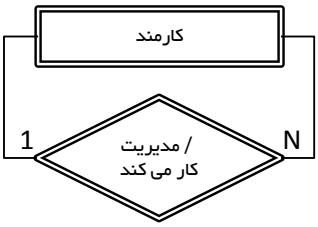
PK

Prof (Pcode , Pname , Dcode)

PK *FK*

تبدیل نمودار ER به جداول

- حالت دوم ، 1 : N



در صورتی که یک موجودیت داشته باشیم.

Employee (emp_no, name,Manager_no)

PK *FK*

تبدیل نمودار ER به جداول

- حالت سوم ، N : M

Course(CourseID ,Course_title,…)

PK

Student (StID ,Stname, …)

PK

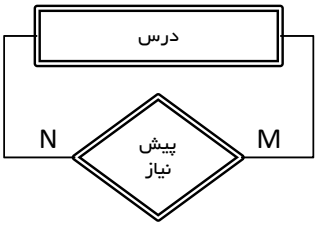
Select(StId , CourseID , …)

PK

تبدیل نمودار ER به جداول

- حالت سوم ، N : M

در صورتی که یک موجودیت داشته باشیم.



Course(CourseID ,Course_title,…)

PK

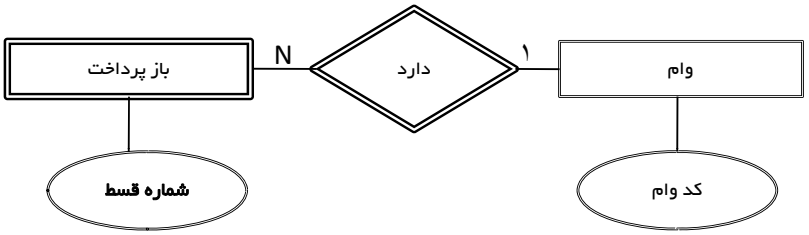
Prereq (CourseID , PreCourseID)

FK *FK*

PK

تبدیل نمودار ER به جداول

- حالت چهارم ، پیاده سازی موجودیت ضعیف



Loan(Loan_code, Amount,…)

صفت ممیزه

Payment(PayNumber, ,Loan_code,Pay_date, Pay_amount)

FK

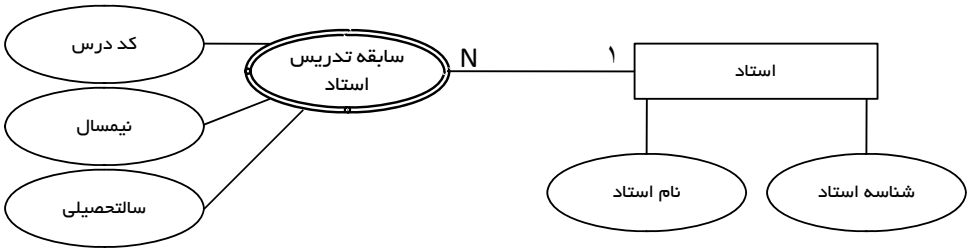
PK

Loan = وام , Payment=بازپرداخت

تبدیل نمودار ER به جداول

- حالت پنجم ، پیاده سازی صفت چند مقداری

اگر تعداد صفات چند مقداری یک نوع موجودیت m باشد ، به m+1 رابطه نیاز است.



Prof (ProfId , ProfName , …)

PK

Prof_Teach (ProfID , CourseID,Semester , Year)

FK

PK

تبدیل نمودار ER به جداول

- حالت ششم ، پیاده سازی ارتباط IS - A

اگر موجودیت مشتق شده با موجودیت پایه صفات غیرمشترک کمی داشته باشند ممکن است فقط یک جدول پیاده سازی شود و فقط یک صفت برای تعیین اینکه کدام زیر نوع مد نظر است به جدول اضافه شود.

مثال: کارمند به عنوان موجودیت اصلی و رئیس، نگهبان و ... بعنوان موجودیت های مشتق شده چون تفاوت فقط سمت می باشد می توان فقط یک موجودیت برای کارمند در نظر گرفته و صفت سمت را به آن اضافه کرد.

تبدیل نمودار ER به جداول

- حالت ششم ، پیاده سازی ارتباط IS - A

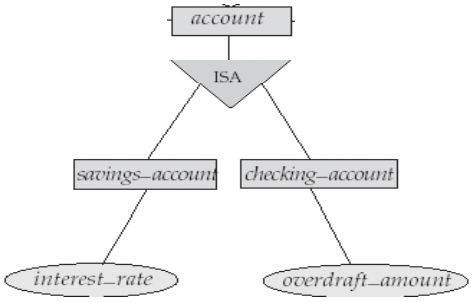
اگر موجودیت پایه با زیر نوع ها صفات مشترک کمی داشته باشد بهتر است برای هر زیر نوع و حتی خود موجودیت اصلی جدول مجزایی پیاده سازی شود.

مثال: کاربر بعنوان موجودیت پایه

دانشجو، استاد و کارمند بعنوان موجودیت مشتق شده

تبدیل نمودار ER به جداول

پیاده سازی ارتباط IS – A

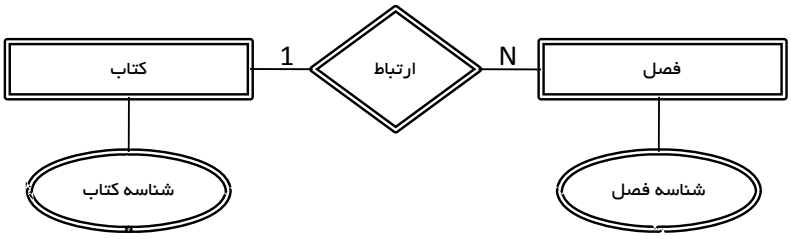


savings_account = (*account_number*, *interest_rate*)

checking_account = (*account_number*, *overdraft_amount*)

تبدیل نمودار ER به جداول

- حالت هفتم ، پیاده سازی ارتباط IS - A Part Of



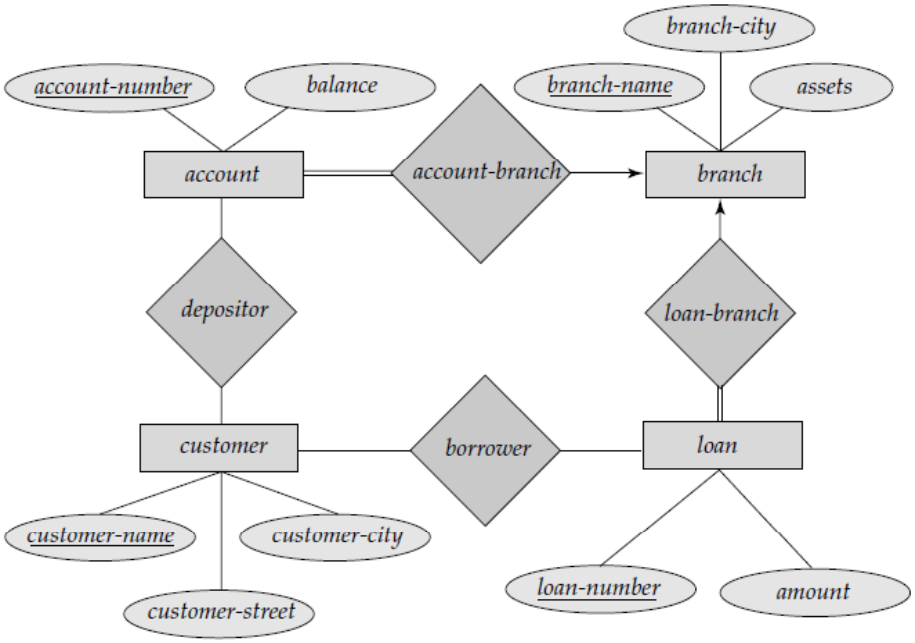
Book (BkId , …)

CK

Chapter (BkId , ChId , …)

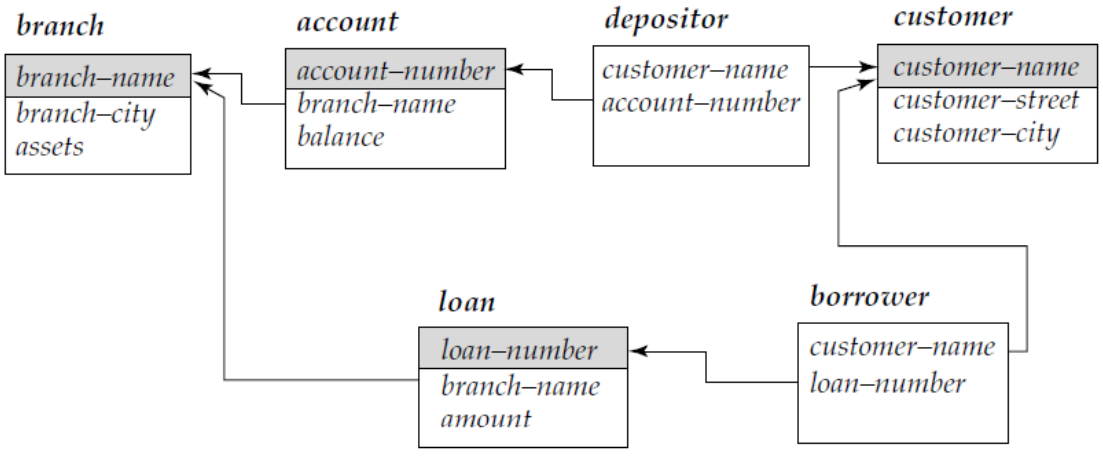
PK

مثال



Loan (وام)
Borrow(وام گرفتن)

پیاده سازی جداول



قواعد جامعیت

- به مجموعه قوانینی که صحت و دقت داده های ذخیره شده در پایگاه داده ها را تضمین می کند ، قوانین جامعیت می گویند.

انواع قوانین جامعیت :

قوانین خاص و کاربردی

میدانی (Domain)

ستونی (Colum)

قوانین عام

قاعده جامعیت موجودیتی (C1)

قابلیت جامعیت ارجاعی (C2)

قواعد جامعیت ـ قوانین خاص و کاربردی

- فقط در یک سیستم خاص و توسط طراح آن پایگاه داده ، تعریف و اعمال می شود و ممکن است در مورد پایگاه داده ی دیگری صدق نکند.

میدانی (Domain):

از طریق تعریف یک دامنه : مجموعه مقادیر مجازی که، یک صفت می تواند به خود بگیرد را دامنه آن صفت گویند. به نوعی می توان گفت DataType هایی که از قبل تعریف شده ؛ مانند Char یا یک دامنه خاص

ستونی (Colum) :

به قواعدی که فقط محدودیت و کنترلی بر روی یک ستون از جدول اعمال می کند، قوانین خاص ستونی گفته می شود. مانند: Balance > 0

قواعد جامعیت ـ قوانین عام

- در همه سیستم ها می بایست اجرا شود.

قاعده جامعیت موجودیتی (C1):

برای تضمین صحت مقدار کلید اصلی می باشد ، یعنی هیچ صفتی تشکیل دهنده ی کلید اصلی ، Null نباشد.

قابلیت جامعیت ارجاعی (C2):

برای تضمین صحت و اعتبار مقدار کلید خارجی است ، طبق این قانون مقدار کلید خارجی می تواند :

Null باشد ، به شرطی که در آن رابطه ، جزئی از کلید نباشد. یا

مقداری معتبر است اگر در جدول مرجع وجود داشه باشد.

نحوه اعمال قواعد c1,c2 در طراحی جداول

- قاعده جامعیت موجودیتی (C1):

از طریق معرفی کلید اصلی در تعریف رابطه

و یا استفاده از گزینه Not Null

- قابلیت جامعیت ارجاعی (C2):

از طریق روش آبشاری (Cascade)

از طریق روش انجام به طور مشروط (Restricted)

از طریق روش هیچ مقدار گذاری (Nullifying)

قواعد جامعیت ـ کاربرد در DBMS

- روش آبشاری (Cascade):

با حذف تاپل هایی از یک جدول مرجع ، تمام تاپل های رجوع کننده به آن حذف شود. (متداول ترین روش)

- روش انجام به طور مشروط (Restricted):

رکورد مرجع را می توان حذف کرد ، به شرطی که ارجاعی به آن نشده باشد.

- روش هیچ مقدار گذاری (Nullifying):

با حذف تاپل های مرجع ، مقدار کلید خارجی در تاپل رجوع کننده Null می شود، به شرطی که کلید خارجی ، جزء کلید اصلی نباشد.

اعمال قواعد جامعیت با زبان SQL

```
create table branch
        (branch_name    char(15),
         branch_city    char(30),
         assets         integer,
         primary key (branch_name),
         check (assets >= 0))

create table account
        (account_number  char(10),
         branch_name     char(15),
         balance         integer,
         primary key (account_number),
         foreign key (branch_name) references branch )
```