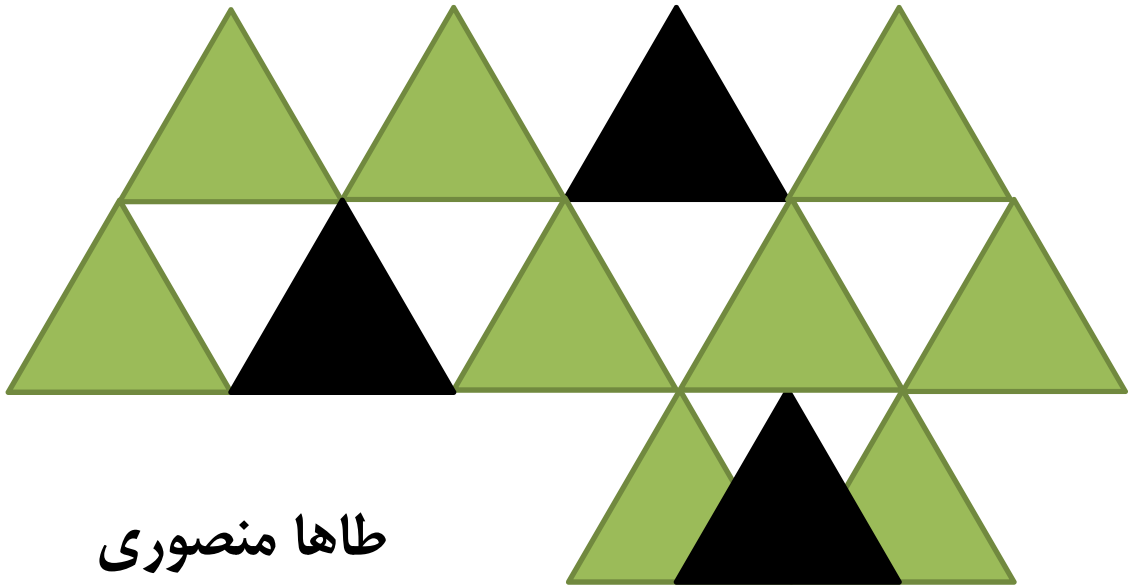
مدیریت پایگاه های داده
فصل سوم- زبان ساختیافته پرس و جو



طاها منصوری

Taha.msi@gmail.com

زبان تعریف داده ها (DDL)

- **این زبان اجازه می دهد که مشخصات زیر تعریف شود:**
  - اسکیما برای هر ارتباط، شامل نوع ویژگی ها
  - محدودیت های یکپارچگی
  - اطلاعات مربوط به مجوزدهی به هر ارتباط
  - افزونه غیر استاندارد SQL نیز مشخصات زیر را تعریف می کند:
    - تعداد زیر مجموعه هایی از هر رابطه که باید نگهداری شود.
    - ساختار فضای ذخیره سازی فیزیکی برای هر رابطه بر روی دیسک

Taha.msi@gmail.com

ساختار ایجاد جدول

❖دستور **SQL** مربوط به ساخت جدول به صورت زیر است:

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

❖ r نام رابطه است.

❖ A_i ها نام ویژگی در اسکیمای رابطه r هستند.

❖ D_i نوع داده ای مربوط به ویژگی A_i است.

❖مثال

```
create table branch
        (branch_name    char(15),
        branch_city     char(30),
         assets          integer)
```


Taha.msi@gmail.com

دامنه انواع داده ای در SQL

- char(n). یک رشته کاراکتری با طول ثابت n که توسط کاربر تعیین می شود.
- varchar(n). یک رشته کاراکتری با طول متغیر حداکثر n که توسط کاربر تعیین می شود.
- int. عدد صحیح که مقدار آن به ویژگی های ماشین وابسته است.
- Smallint. عدد صحیح کوچک که مقدار آن به ویژگی های ماشین وابسته است.
- numeric(p,d). عدد اعشاری با مقدار ثابت که توسط کاربر مشخص می شود.
- real, double precision. مقادیر اعشاری وابسته به ویژگی های ماشین
- float(n). مقدار اعشاری با طول تعیین شده توسط کاربر

Taha.msi@gmail.com

محدودیت های یکپارچگی بر روی جداول

- یکتایی
- مقادیر مجاز
- عدم تخصیص ارزش های تهی
- کلید اصلی
- مثال

```
create table branch
        (branch_name       char(15),
         branch_city char(30) not null,
         assets                integer,
         primary key (branch_name))
```


Taha.msi@gmail.com

حذف و اضافه چندگانه ها

- جدول های ساخته شده
- با دستور زیر می توان یک چندگانه به جدول:

- insert into *account*
  values ('A-9732', 'Perryridge', 1200)

- اضافه کردن در صورت نقض محدودیت ها، خطا می دهد.
- با دستور زیر می توان همه سطرها را حذف کرد:

- delete from *account*

Taha.msi@gmail.com

حذف یا تغییر جدول و محدودیت هایش

- دستور Drop Table همه اطلاعات مربوط به ارتباط مورد نظر را از پایگاه داده حذف می کند.
- دستور Alter Table برای افزودن ویژگی به رابطه مورد استفاده قرار می گیرد:
- alter table *r* add *A D*
  - r یک رابطه، A ویژگی و D نوع داده ای ویژگی است.
- همه چندگانه های موجود در رابطه، مقدار تهی به ویژگی افزوده شده تخصیص می دهند.
- دستور Alter Table همچنین برای حذف ویژگی از رابطه نیز مورد استفاده قرار می گیرد:
- alter table *r* drop *A*
  - *حذف ویژگی در بسیاری از پایگاههای داده ای پشتیبانی نمی شود.*

Taha.msi@gmail.com

ساختار پایه جستجو

- یک پرس و جوی معمولی به این شکل است:

select $A_1, A_2, ..., A_n$
from $r_1, r_2, ..., r_m$
where P

- Ai نشان دهنده ویژگی است
- Ri نشان دهنده رابطه است.
- P نیز نشان دهنده شرط است.
- خروجی یک پرس و جو یک رابطه است

Taha.msi@gmail.com

دستور انتخاب (select)

- دستور select ویژگی های یک رابطه بعنوان نتیجه لیست می کند.
- مثال نام همه شعبه ها را در رابطه وام پیدا کنید:

```
select branch_name
    from loan
```

- نکته قابل توجه این است که دستورات SQL حساسیتی به بزرگی و کوچکی حروف ندارند
- SQL امکان روابط چندگانه را نیز در اختیار می گذارد.
- برای ازبین بردن ارزش های تکراری از کلمه distinct استفاده می شود.

```
select distinct branch_name
    from loan
```

- علامت * نیز برای انتخاب همه ویژگی ها مورد استفاده قرار می گیرد:

```
select *
    from loan
```


Taha.msi@gmail.com

بخش شرط (where)

- Where شروطی را تعیین می کند که نتایج باید آنها را ارضا کنند.
- شرایط را می توان با عملگرهای and, or, not باهم مرتبط کرد.
- از عملگرهای in و not in برای اطمینان حاصل کردن از وجود یک ارزش در یک مجموعه می توان استفاده کرد.
- مثال؛ شماره وام هایی را از رابطه وام پیدا کنید که نام شعبه آنها Perryridge یا lumma بوده و مقدار آنها بیشتر از ۱۲۰۰ دلار است:

```
Select loan_number form loan
Where branch_name="Perryridge"
or branch_name="lumma"
and amount>1200
```

یا

```
Select loan_number from loan
where branch_name in ("Perryridge", "lumma")
and amount>1200
```


Taha.msi@gmail.com

بخش مقصد(from)

- بخش from، ارتباطاتی را که پرس و جو بر روی آنها صورت میپذیرد مشخص می کند.
- مرتبط با محصول دکارتی در جبر رابطه ای است.
- مثال، محصول دکارتی وام و قرض گیرنده را حساب کنید:
- Select * from borrower, loan
- نام، شماره وام، و میزان وام را برای همه مشتریانی پیدا کنید که در شعبه Perryridge وام گرفته اند:

```
select customer_name, borrower.loan_number, amount
        from borrower, loan
        where   borrower.loan_number = loan.loan_number
and     branch_name = 'Perryridge'
```


Taha.msi@gmail.com

عملیات تغییر نام

- SQLبا استفاده از دستور as اجازه می دهد که نام روابط و ویژگی ها تغییر یابد.
- مثال: نام، شماره وام، و میزان وام همه مشتریان را بدست آورده و به جای شماره وام، آی – دی وام را قرار دهید:

```
select customer_name, borrower.loan_number as
loan_id, amount
from borrower, loan
where borrower.loan_number = loan.loan_number
```


Taha.msi@gmail.com

مرتب کردن خروجی انتخاب

- با استفاده از order by و افزودن desc برای نزولی و asc صعودی، می توان لیست خروجی انتخاب را بر اساس ویژگی یا ویژگی هایی مرتب کرد:

- select distinct *customer_name*
  from *borrower, loan*
  where *borrower loan_number = loan.loan_number*
  and
  branch_name = 'Perryridge'
  order by *customer_name*

Taha.msi@gmail.com

عملیات روی مجموعه ها

- دستورات union, intersect, except همان عملیاتی را انجام می دهند که در جبر رابطه ای نیز انجام می دهند.
- همه مشتریانی را بیابید که وام دارند، حساب دارند یا هردو را دارند:

(select *customer_name* from *depositor*)
union
(select *customer_name* from *borrower)*

- همه مشتریانی را بیابید که هم وام و هم حساب دارند

(select *customer_name* from *depositor*)
intersect
(select *customer_name* from *borrower)*

- همه مشتریانی را بیابید که حساب دارند اما وام ندارند:

(select *customer_name* from *depositor*)
except
(select *customer_name* from *borrower)*

Taha.msi@gmail.com

توابع تجمیعی

- این توابع برای روی چند مجموعه از ارزش های ستونی از یک رابطه عمل کرده و یک ارزش باز می گردانند:
  - Avg
  - Min
  - Max
  - Sum
  - Count
- در صورت نیاز به بخش بندی از عبارت group by استفاده می شود:

```
select branch_name, count (distinct customer_name)
        from depositor, account
        where depositor.account_number = account.account_number
        group by branch_name
```

برای افزودن شرط بر روی خروجی توابع تجمیعی نیز از عبارت having استفاده می شود:

```
select branch_name, avg (balance)
        from account
        group by branch_name
        having avg (balance) > 1200
```


Taha.msi@gmail.com

تغییر پایگاه داده–حذف

- از دستور delete به همراه بخش where برای حذف سطرهای مورد نظر از رابطه استفاده می شود:

- delete from *account*
  where *branch_name* = 'Perryridge'

- یا

- delete from *account*
  where *branch_name* in (select *branch_name*
  from *branch*
  where *branch_city* = 'Needham')

Taha.msi@gmail.com

تغییر پایگاه داده–اضافه

- با استفاده از دستور insert مقادیر جدید به روابط اضافه می شود:
- insert into *account*
  values ('A-9732', 'Perryridge', 1200)

Taha.msi@gmail.com

تغییر پایگاه داده–اضافه

- با استفاده از دستور insert مقادیر جدید به روابط اضافه می شود:
- insert into *account*
  values ('A-9732', 'Perryridge', 1200)

Taha.msi@gmail.com

تغییر پایگاه داده–بروز رسانی

- با استفاده از دستور update می توان مقادیر ویژگی های موجود در یک رابطه را بروز رسانی کرد:

- update *account*
  set *balance* = *balance* * 1.06
  where *balance* > 10000

Taha.msi@gmail.com