پایگاه داده پیشرفته

محمد داورپناه جزی

ترم دوم ۹۴-۹۳

گروه مهندسی کامپیوتر و فناوری اطلاعات

دانشگاه صنعتی فولاد

فصل سوم: همزمانی, ۳-۱ مسئله تداخل

- **بحث فصل قبل:** اجرای درست یک تراکنش به تنهایی, این فصل: اجرای موازی
- **نتیجه:** اجرای یک تراکنش درست روی یک DB درست ← یک DB درست جدید
- **شرط حصول نتیجه فوق:** اجرای تراکنش به صورت تنها و مستقل
- **استفاده اشتراکی از DB:** اجرای درست هر تراکنش به تنهایی ولی عدم اجرای درست در صورت اجرای همزمان, بردن DB به یک وضعیت نادرست در صورت اجرای همزمان
- **مثال مناسب:** موضوع **lost update**, ش ۳-۱ ص ۸۴, مثلا برای رزرو بلیط یا . . .
- **خلاصه:** نیاز به مکانیزم کنترل همزمانی و پیشگیری از موارد اینچنینی

سه روش به شرح زیر با توجه به مثال ش ۳-۱

- ممانعت از خواندن توسط B در t2, چون A قبلا خوانده و ممکن است تغییر دهد
- ممانعت از اصلاح توسط A در t3, چون B هم خوانده و ممکن است تغییر دهد
- ممانعت از اصلاح توسط B در t4, چون A خوانده و تغییر هم داده
- **خلاصه:** دو حالت اول **locking** و حالت سوم **timestamping**

فصل سوم: همزمانی, ۳-۲ قفلهای انحصاری

- **منطق قفل:** تکنیکی برای تنظیم و کنترل دسترسیها به یک شی مشترک
- **شی مشترک:** یک واحد قابل قفل, فرض این فصل رکورد به عنوان واحد قابل قفل
- **مسئول عملیات قفل گذاری و رفع قفل:** مدیر قفل یا **lock manager**
- **روند عملیات:** ۱) درخواست یک تراکنش به **LM**, برای اعمال قفل روی واحد
  ۲) حفظ قفل در قالب یک کنترل بلاک حاوی شناسه های رکورد (واحد) و تراکنش
  ۳) اطمینان (تقریبی) تراکنش از عدم انجام دسترسی همزمان به رکورد (بحث بعد)
  ۴) انجام عملیات مورد نظر توسط تراکنش
  ۵) درخواست تراکنش به **LM**, برای رفع قفل از رکورد
- **خلاصه:** با وجود قفل روی **R** برای تراکنش **T**, دادن گارانتیهای خاص به **T** در طول قفل, ماهیت گارانتیها بسته به نوع قفل
- **معمول ترین قفل:** قفل انحصاری (**Xlock**), بعد از اعمال قفل **X** روی **R** توسط **T**, عدم امکان اعمال قفل دیگر توسط **T'** مگر پس از رفع قفل مزبور

فصل سوم: همزمانی, ۳-۲ قفلهای انحصاری (ادامه)

- **پروتکل PX**: مورد نظر به منظور رفع مسئله **lost update** ش ۳-۱ ص ۸۴
- **روند عمل**: لزوم شروع با **xfind** کردن **R** برای به روز رسانی آن, ارائه در صورت امکان و اعمال قفل, و گرنه منتظر شدن تراکنش برای **R**
- **نکته**: لزوم گارانتی **DBMS** برای سرویس تراکنشهای معطل به دلیل فوق, در آینده, برای پیشگیری از انتظار تا ابد (شرط **livelock**)
- **قفل ضمنی**: امکان اعمال مراحل فوق به طور ضمنی در دستور **find**
- **مثال**: اعمال **PX** روی ش ۳-۱ و حصول ش ۳-۲ ص ۸۷, حالت اول از سه حالت ص ۲

آزادسازی: اعلام آن با دستور **xrelease** , ش ۳-۲

- **حذف رکورد**: لزوم انجام **Xfind** قبل از حذف مشابه به روز رسانی, مثال حذف به جای اصلاح در ش ۳-۱ ص ۸۴
- **ایجاد رکورد**: نیاز به قفل مشابه روی رکورد درحال ساخت مشابه دو حالت دیگر

فصل سوم: همزمانی, ۳-۳ ترتیب پذیری

- **فرضیات:** a) فرض بر درستی همه تراکنشهای منفرد و حفظ یکپارچگی DB در صورت اجرا به تنهایی

  b) درستی عملیات و حفظ یکپارچگی DB در اثر هر ترتیب اجرای مجموعه تراکنش به صورت T = {T1, T2, . . ., Tn} هر یک به طور منفرد

  c) امکان حصول نتایج متفاوت در اثر دو ترتیب متفاوت, اصل درستی و یکپارچگی DB

  d) اجرای غلط در ش ۳-۱ (ترتیب ناپذیر), اجرای درست در ش ۳-۲ (ترتیب پذیر)

  e) احتساب ترتیب پذیری به عنوان ملاک رسمی درستی تراکنشها, یک ترتیب متداخل تراکنشها درست است اگر ترتیب پذیر باشد

- **تعریف:** مجموعه تراکنشهای متداخل (interleaved) را ترتیب پذیر گویند اگر نتیجه اجرای آن دقیقا مشابه نتیجه یک اجرای ردیفی آن باشد.

  ترتیب پذیر به معنی به ترتیب اجرا شدن نیست

- **نکته:** تنها ترتیب ممکن در یک سیستم یک حالت ترتیب پذیر ← سیستم همیشه درست

فصل سوم: همزمانی, ۳-۳ ترتیب پذیری (ادامه)

- **مفهوم کلی:** E یک اجرای متداخل از یک مجموعه تراکنش {T1, T2, . . ., Tn} باشد, E ترتیب پذیر است اگر نتیجه آن معادل نتیجه یک اجرای ترتیبی S از این تراکنشها باشد. S یک ترتیب از E است و لزوما یگانه نیست.
- **مفهوم جزئی:** برای هر Ti و Tj در S که Ti قبل از Tj باشد, در E هم باید نتیجه مثل این باشد.

یعنی برای هر A و B در یک مجموعه تراکنش ترتیب پذیر, یا A قبل از B است (A نتایج B را می بیند) یا برعکس, اگر A رکوردهای R1 تا Rn را اصلام می کند, B یا همه را قبل از اصلام می بیند یا بعد از اصلام ولی نه مخلوطی از آنها, اگر وضعیت A قبل از B یا برعکس بر قرار نباشد, اجرا ترتیب پذیر نیست و غلط است.

- **حالت خاص:** A و B بدون ارتباط و با ترتیب S1, جا به جا کردن آنها ایجاد ترتیب جدید S2
- **یادآوری:** ترتیب پذیر به معنی به ترتیب اجرا شدن نیست

فصل سوم: همزمانی, ۳-۳ ترتیب پذیری (ادامه)

- **مثال:** ش ۳-۳ ص ۸۹, با سه ترا کنش متداخل با زمان اجرای مشخص (کروشه ها)

  A قبل از B به طور منطقی, B قبل از C به طور منطقی, C قبل از A به طور فیزیکی

  A رکورد R2 را قبل از اصلاح B می بیند پس A قبل از B است

  B رکورد R1 را قبل از اصلاح C می بیند پس B قبل از C است

  اجرای C قبل از A است!

  اجرا ترتیب پذیر است به صورت: A→B→C

- **فرضیات:** قفل B روی R1 باید قبل از t2 برداشته شود تا C کار کند (تضاد با 2PL) قفل A روی R2, خود به خود در t5 برداشته می شود جون تمام شده

  R1 و R2 باید مستقل از هم باشند, اصلاح R2 در t6 نباید وابسته به مقدار R1 در t1 باشد و گرنه قفل B روی R1 می ماند و C مشکل پیدا می کند, تضاد با فرض بالا

فصل سوم: همزمانی, ۳-۴ بن بست

- **موضوع:** با وجود حل مشکل **lost update** با پروتکل **px**, امکان بروز بن بست هنوز
- **مثال:** ش ۳-۴ ص ۹۱, وقوع بن بست بین **A** و **B** در زمان **t4**
- **پیشنهاد:** ایجاد یک ترتیب کلی روی اشیای قفل شونده, **Ti** بخواهد **X** را قفل کند وقتی **Y** را دارد, **Y** بعد از **X** است, **X** قفل نمی شود که بن بست نشود.
- **در پایگاه داده:** امکان ندارد چون شناسایی **X** ممکن است وابسته به اجرای **Y** باشد
- **تفاوتهای DB با سایر محیطها:**

حجم بسیار زیاد اشیای قابل قفل و تغییر پذیری بالا

رجوع به رکوردها (اشیا) از طریق محتوا (نه آدرس) و در زمان اجرا, امکان تشخیص دو رجوع به یک رکورد در آن زمان

دینامیک بودن محدوده محدوده مورد قفل ترا کنشها

نتیجه: لزوم آمادگی سیستم برای پذیرش و حل مسئله بن بست

فصل سوم: همزمانی, ۳-۴ بن بست (ادامه)

- **روش کار:** ساخت گراف انتظار, ترا کنشها نود و انتظارها لبه, امکان ایجاد سیکل بین دو یا چند نود, نتیجه بروز بن بست, مثال . . .
- **پیشگیری:** بعد از هر انتظار برای کسب قفل, نتیجه خوب ولی هزینه بالا

  آزمون دوره ای, هزینه پایین, قابل تنظیم, امکان از دست دادن موارد و وقوع بن بست

  timeout, قطع ترا کنش درصورت عدم فعالیت در مدت خاص, مشابه قبلی, کم هزینه
- **رفع:** نیاز به یک قربانی که **RB** شود, نه لزوما منجر به بن بست, مثلا با شروع دیرتر, دارای کمترین منابع قفل شده, با تعداد اصلاح کمتر, . . .
- **ملاحظات اجرایی:** لزوم **RB** کردن همه اصلاحات و رفع قفلها, در سطح سیستم بودن موضوع, لزوم دسترسی مدیر **DL** به منابع برای رفع قفل و تعامل با سیستم عامل
- **تکلیف قربانی (مقصر نبوده):** پیغام سیستم و امکان اجرای مجدد توسط کاربر (مثل **system R**), امکان اجرای مجدد ترا کنش به طور اتوماتیک (مثل **IMS**)

فصل سوم: همزمانی, ۳-۵ بررسی مجدد lost update

- موضوع: وجود عواقب مرتبط با همزمانی در صورت RB کردن تراکنشها
- نمونه اول: ش ۳-۵ ص ۹۳, استفاده B از نتیجه اصلاح R توسط A ولی لغو A وابستگی به اصلاحات CMT نشده, تغییرات CMT نشده, داده کثیف, (ترتیب ناپذیر)

نمونه دوم: ش ۳-۶ ص ۹۴, اصلاح مجدد B روی اصلاح A که RB می شود, وخیم تر! راه حل: ممنوع کردن به هنگام سازی نتیجه عمل CMT نشده, اجازه ندادن XRLS در t3, امکان این کار فقط بعد از ET, اصلاح پروتکل PX, انتظار B تا بعد از t6

- پروتکل PXC: لزوم شروع با xfind کردن R برای به روز رسانی آن, ارائه در صورت امکان و اعمال قفل, و گرنه منتظر شدن تراکنش برای R حفظ Xlock تا زمان خاتمه تراکنش با CMT یا RB, کافی برای پیشگیری از LU
- نتیجه: عدم نیاز به XRLS, رفع تمام قفلها ضمن CMT و RB
- قابل تحمل بودن نمونه اول در بعضی شرایط برای گزارش آماری, عدم نیاز به دقت ..۱% (MSW), عدم انتظار برای قفل, عدم دقت, نمونه دوم اکیدا غیرقابل قبول

فصل سوم: همزمانی, ۳-۶ قفلهای اشتراکی

- **موضوع:** لزوم حفظ قفل حتی در صورت عدم اعمال اصلاح توسط تراکنشها
- **نمونه اول:** ش ۳-۷ ص ۹۷, فقط استفاده A از رکورد I و نه اصلاح, نتیجه غلط در A
  لزوم اعمال قفل روی رکورد I توسط A, مجوز فقط خواندن برای دیگران, **Slock**
  روند قفل: قفل S توسط تراکنش T روی رکورد R, امکان اعمال همین قفل توسط دیگران ولی نه قفل X, مگر بعد از رفع همه قفلهای S
  وجود دستور **Sfind** متناظر با **Xfind**, رفتن به انتظار در صورت عدم امکان
  دستورهای **Srelease** و **Sreleaseall** برخلاف عدم وجود **Xrelease**
- **نمونه دوم:** اصلاح نمونه اول در ش ۳-۸ ص ۹۷, ولی حاوی بن بست
  حذف یکی از تراکنشها برای ایجاد امکان خاتمه دیگری
  ش ۳-۹ ص ۹۸ حاوی ترکیبات مجاز قفل و بدون قفل, کمتر کردن قفل X با پروتکل **PS**
- **پروتکل PS:** **Sfind** تراکنش در شروع به هنگام سازی, تبدیل قفل S به X و **Updx**
- پروتکل **PSC** هم به جای **PXC** بهتر عمل می کند (حفظ قفل X تا خاتمه کار)

فصل سوم: همزمانی, ۳-۶ قفلهای اشتراکی (ادامه)

- **امکان بروز بن بست:** ش ۳-۱۰ ص ۹۹ تبدیل یافته ش ۳-۱ با قفل S, حالت b ص ۸۴
- **ارتقای قفل:** ش ۳-۹ ارتباط قفلها در حالت دو تراکنش مستقل و مجزا, حالت جدید ارتقای قفل یک تراکنش روی یک شی خاص, مثل **sfind** سپس **updx**

  لزوم تعریف قفلهایی غیر از S و X برای این موارد

  مولفه های قفل: تعریف ساده قفل **<lock object ID, transaction ID>**

  به اضافه سه مولفه زیر:

  ۱) تعداد دفعاتی که تراکنش فعلی این نوع قفل را درخواست کرده

  ۲) تعداد دفعاتی که سیستم از طرف تراکنش این نوع قفل را درخواست کرده (مثل احتساب **sfind** به جای **find** توسط سیستم)

  ۳) قوی ترین نوع قفلی که قفل فعلی تحت آن اخذ شده است

  نتیجه: یک شی تحت قفل است اگر یکی از شمارنده های بالا غیر صفر باشد, با شرط افزایش شمارنده با اخذ قفل و کاهش آن با **CMT** یا **RB**

فصل سوم: همزمانی, ۳-۷ قفلهای به روز رسانی

- **پروتکلهای قبلی:** در PS امکان همزمانی بیشتر ولی احتمال بروز بن بست در PX امکان همزمانی محدودتر و احتمال بن بست کمتر
- **نتیجه:** مهمترین منبع بن بست درخواستهای ارتقای قفل است (ش ۳-۱۰)
- **راه حل:** تعریف قفل U برای وقتی که تراکنش بخواهد امکان انجام به هنگام سازی را اعلام کند, ماتریس سازگاری در ش ۳-۱۱ ص ۱۰۱, نیاز به پروتکل جدید
- **پروتکل PU:** برای تراکنشی که قصد به هنگام سازی R را دارد نخست باید Ufind R کند تا قفل U بگیرد, بعد برای بروز رسانی باید UPDX R کند تا قفل تقویت شود به X به اضافه تبدیل روتکل PXC به PUC (حفظ قفل X تا خاتمه کار)

  نتیجه: پیشگیری از بن بست با تبدیل Xfindها به Ufind در شکل ۳-۱۰, امکان همزمانی بیشتر, گارانتی امکان به هنگام سازی با وجود قفل U
- **کاربرد:** system R دارای PSC, IMS دارای PUC و DBTG دارای PXC یا PSC

  کاربرد یک پروتکل (PSC, PUC و PXC) در هر سیستم

فصل سوم: همزمانی, ۳-۸ قفل کردن دو فازی

- **آنچه گذشت:** لزوم حفظ قفل به هنگام سازی تا پایان کار تراکنش
  امکان آزادسازی سایر قفلها قبل از خاتمه کار در صورت لزوم
  خطر بروز نتایج غلط درصورت آزادسازی یک قفل و اخذ قفل دیگر به دلیل امکان وجود تراکنش همزمان و بروز ترتیب ناپذیری
- **قضیه (دوفازی)!:** اگر همه تراکنشهای یک مجموعه از دو قاعده زیر پیروی کنند
  الف) قبل از شروع به کار روی هر شی, نخست قفلی (از هر نوع) روی آن اعمال کند
  ب) بعد از رها سازی یک قفل, قفل دیگری درخواست نکند
  در این صورت همه اجراهای متداخل مجموعه تراکنش مزبور ترتیب پذیر خواهد بود
- **و گرنه:** لزوم سازش, امکان ترفیع همزمانی, احتمال بروز نتایج ناخواسته
- **تعریف:** هر تراکنش تابع قواعد فوق (پروتکل 2PL) دو فازی گفته می شود, فاز ترفیع (growing) برای اخذ قفلها و فاز تقلیل(shrinking) برای تقلیل و رها سازی قفلها
- **نتیجه:** اگر همه تراکنشها دو فازی باشند, آنگاه همه اجراهای آنها ترتیب پذیرند

فصل سوم: همزمانی, ۳-۸ قفل کردن دو فازی (ادامه)

- **برداشت غلط:** تراکنشها باید دوفازی باشند تا اجراهای متداخل ترتیب پذیر باشند
  شرط کافی است ولی برای گارانتی ترتیب پذیری لازم نیست! توجه به مثال زیر
- **مثال:** ص ۱.۳ و ش ۳-۱۲ ص ۱.۴, تفاوت در اجرای **AB** و **BA**
  اجرای ش ۳-۱۲ معادل **AB** پس ترتیب پذیر اگرچه غیر **2PL**
  اعمال **2PL**, عدم امکان **Xfind** در **t7** و شکست, عدم امکان اجرا اگرچه ترتیب پذیر
- **در صورت قبول:** شرایط امن برای آزادسازی قفل روی **F**, عدم تاثیر تراکنشهای همزمان با **A** روی آن, امکان تعریف در طراحی ولی عدم امکان کنترل در زمان اجرا
- **مثال :** ص ۱.۴ و ش ۳-۱۳ ص ۱.۵, تفاوت در اجرای **AC** و **CA**
  اجرای ش ۳-۱۳ معادل هیچکدام, پس ترتیب ناپذیر و نادرست به دلیل غیر **2PL**
  اعمال **2PL**, عدم امکان **Xfind** در **t7** و جلوگیری از اجرای غلط
- **نتیجه:** سپردن خود به قضا و قدر در شرایط این گونه

فصل سوم: همزمانی, ۳-۸ قفل کردن دو فازی (ادامه)

- **مشکل بیان قضیه:** نیاز به اعمال قفل روی عدم وجود یک شی در بعضی موارد
- **مثال:** ص ۱۰۶ و ش ۳-۱۴ ص ۱۰۷, تشابه نتایج در اجرای **AB** و **BA**

  اجرای ش ۳-۱۳ معادل هیچکدام, پس ترتیب ناپذیر و نادرست

  لزوم جلو گیری از ایجاد **F** در **B** مادامی که **A** فکر می کند **F** وجود ندارد

  اصطلاح **phantom** یا شبح برای **F**
- **نیاز:** لزوم امکان اعمال قفل توسط تراکنش **A** روی عدم وجود **F** در **t1** و جلو گیری از ایجاد **F** در **t3** توسط تراکنش **B**
- **راهکار :** فرض بر وجود فیزیکی همه فیلدها, افزودن نشانگر برای اعلام وجود منطقی فیلدها وقتی **insert** شوند, تبدیل **find** به **sfind** در ش ۳-۱۴, اعمال قفل **F** در **t1** برای تراکنش **A** و انتظار تراکنش **B** در **t3**

مثال دیگر: عدم رعایت **2PL** در ش ۳-۳ ص ۸۹ ولی ترتیب پذیر

فصل سوم: همزمانی, ۳-۹ جلو گیری از بن بست

- **واقعیت:** سیستمهای موجود عموما ایجاد کننده بن بست تا پیش گیرنده آن
- **در سیستم متمرکز:** تحمل بن بست کم هزینه تر از پیشگیری از آن
- **در سیستم توزیع شده:** لزوم پیشگیری از بن بست به دلیل اهمیت موضوع
- **روشها :** Transaction Scheduling یا برنامه ریزی تراکنشها,
  - Request Rejection یا رد ردخواست
  - Transaction Retry یا تکرار تراکنش
  - Timestamping یا برچسب زمانی
- **۱) برنامه ریزی تراکنشها:** برنامه ریزی اجرای تراکنشها به ترتیبی که از بروز بن بست جلو گیری شود, نیاز به اطلاع از نحوه اجرای تراکنشها قبل از اجرای آنها
- **مشکل:** عدم امکان تعیین این موارد, مثال انتقال پول بین دو حساب, واحد قالب قفل کل مجموعه رکوردها, اعمال قفلها در شروع برنامه, غیر قابل اجرا

فصل سوم: همزمانی, ۳-۹ جلوگیری از بن بست (ادامه)

- **۲) رد درخواست:** رد درخواست قفل در صورتی که منجر به بن بست شود, منتظر شدن تراکنش و ایجاد سیکل در گراف انتظار با افزودن لبه مرتبط با این درخواست
- **خصوصیات:** بهتر از تحمل بن بست و قربانی کردن یک پروسس, شانس کمی انتظار و درخواست مجدد برای تراکنش, امکان tidy up و اعلام وضعیت به کاربر
- **۳) تکرار تراکنش:** برای محیطهای توزیع شده, امکان استفاده در محیط متمرکز
- **دو روش : wait die و wound wait**, جلوگیری از بروز سیکل در گراف انتظار, تخصیص برچسب زمانی به هر تراکنش (زمان شروع), عدم شروع دو تراکنش همزمان
- **وضعیت:** درخواست A برای ایجاد قفل روی منبعی که توسط B قفل شده
- در **wait die:** صبر کردن A اگر پیرتر از B است و مردن (RB) اگر جوانتر است
- در **wound wait:** صبر کردن A اگر جوانتر از B است و زخمی (RB) کردن اگر پیرتر

ارتباط wait و wound به پیرتر بودن A از B, حفظ برچسب زمانی اولیه برای اجرای مجدد,

WD صبر کردن پیرترها, WW صبر کردن جوانترها, عدم ایجاد حلقه, ش ۳-۱۵ و ۳-۱۶

- **۴) برچسب زمانی:** توضیح در بخشهای بعدی, عدم وجود تکنیک قفل در این روش, قفل نشدن هیچ شیئی و در نتیجه عدم امکان بروز بن بست

فصل سوم: همزمانی, ۳-۱ دانه بندی قفل

- **دانه بندی قفل:** حجم داده های مورد قفل, کل DB, یک جدول, یک رکورد, یک فیلد
- **دانه بندی ریز:** همزمانی بیشتر, بالاسری بیشتر, تعداد قفل بیشتر, احتمال DL بالاتر
- **دانه بندی درشت:** همزمانی, بالاسری و تعداد قفل کمتر, احتمال DL پایین تر
- **مثال :** قفل DB, همزمانی صفر, فقط یک قفل, بدون DL
- **قفل منطقی و فیزیکی :** پیاده سازی مفهوم منطقی (نرم افزاری) قفل در عمل با قفل یک بلوک داده سخت افزاری, استفاده از همان روش قفل گذاری, اصول در ادامه
- **اصل اول پیاده سازی:** سیستم می تواند به طور امنی فضایی وسیع تر از آنچه خواسته شده را قفل نماید, یکپارچگی به خطر نمی افتد ولی همزمانی چرا, (مثال Access), احتمال بروز DL
- **اصل دوم پیاده سازی:** سیستم می تواند قفلی را بیشتر از مدت زمانی که خواسته شده است, حفظ نماید
- نکته: اخذ اینگونه قفلها در برنامه و لغو آنها با خاتمه برنامه و نه با CMT یا RB

فصل سوم: همزمانی, ۳-۱۰ دانه بندی قفل (ادامه)

- آزاد سازی در CMT:

  نیاز به مکانیزمی برای اخذ مجدد قفل توسط برنامه بعد از شروع
  مشکل بودن کار, منجر به انتظار در صورت گرفتن رکورد توسط دیگری

- نوع قفل: Ulock دقیق ترین حالت قفل

  مناسب برای رکورد ولی نامناسب برای جدول
  بر مبنای تعریف U احتمال نیاز به تبدیل به Xlock در ادامه
  اشکال در تبدیل U به X که برای محدوده قفل کوچکتر مناسب است
  امکان تاخیر زیاد در صورت قفل جدول با X

فصل سوم: همزمانی, ۳-۱۱ قفل قصدی یا intent

- **موضوع:** نیاز تراکنش T به جدول (baseset) B به صورت stable
  عدم امکان اعمال تغییر همزمان رکوردی از B توسط تراکنش دیگر
  عدم تغییر B توسط تراکنش T پس امکان اخذ B با یک قفل S, ولی
  لزوم اطمینان از عدم وجود قفل X روی هیچ رکورد B برای سایر تراکنشها, وگرنه . . .
- **مشکل:** عدم امکان بررسی وجود قفل X روی رکوردهای B
  عدم امکان تست قفلهای X که روی رکوردهای B نباشند
- **پروتکل قفل قصدی:** مبتنی بر مفهوم جدول و رکورد, وجود یک سطح دسترسی خاص برای هر تراکنش روی جدول جهت ایجاد امکان دستیابی به رکوردهای جدول
  سطح دسترسی خاص = اعمال قفل خاص, توضیح عملی روی UDL برای دقت بیشتر
- **در UDL:** تعریف سطوح دسترسی در قالب PROCLEVEL و SHARELEVEL
  اولی به معنی آنچه می خواهد انجام دهد و دومی آنچه می تواند تحمل کند, ص ۱۱۲

فصل سوم: همزمانی, ۳-۱۱ قفل قصدی یا intent (ادامه)

- **حل مشکل قبلی:** اعلام تراکنش T روی جدول B به صورت زیر
  SHARELEVEL(REF) به معنی قبول دسترسی دیگران ولی نه اصلاح
  PROCLEVEL(REF) به معنی فقط قصد دسترسی توسط T و نه اصلاح
- **نحوه عمل سیستم:** (وضعیت وجود قفلها روی رکوردهای B).
  a) آگاهی سیستم از تمام تراکنشهای در حال کار روی B
  b) آگاهی سیستم از SHLVL و PRLVL همه تراکنشها
  c) ارتباط دقیق و تعریف شده SHLVL ها و PRLVL ها طبق جدول ۳-۱۷
- **مفهوم جدول:** تراکنش T در سطر بالا و 'T در ستون چپ, PRLVL بالا و SHLVL زیر
- **نیاز به نوع قفل جدید:** عدم پشتیبانی برخی حالات با توجه به جدول ۳-۱۸, نیاز جدید
  معرفی سه قفل IS, IX و SIX برای اعمال روی جداول
- **قفل IS:** قصد T برای گذاشتن قفل S روی رکوردهای جدول برای گارانتی پایداری
- **قفل IX:** شبیه قبلی ولی احتمال اصلاح بعضی رکوردها توسط T با اعمال قفل X روی آنها

فصل سوم: همزمانی, ۳-۱۱ قفل قصدی یا intent (ادامه)

- **قفل S**: تحمل T برای خواندنهای همزمان ولی نه اصلاح همزمان در B.
  عدم اصلاح توسط T در B
- **قفل SIX**: تلفیق S و IX, تحمل T برای خواندنهای همزمان ولی نه اصلاح در B,
  احتمال اصلاح بعضی رکوردها توسط T با اعمال قفل X روی آن رکوردها
- **قفل X**: عدم تحمل T برای حتی خواندنهای همزمان در B, احتمال اصلاح رکوردهای T
- **سازگاری قفلها**: وضعیت قفلهای سطح جدول در ش ۳-۱۹, مشابه ش ۳-۹ و تکمیل ۳-۱۷
- **سطوح دسترسی**: تکمیل جدول ۳-۱۸ در ش ۳-۲.۰ برای پیاده سازی سطوح دسترسی
- **نتیجه, اعمال قفل X و S در سطح جدول**: عدم نیاز به قفل در سطح رکورد
- **اعمال قفل I روی جدول**: اعمال قفل روی رکوردها به طور اتوماتیک به شرح زیر

برای IS, اعمال قفل S روی رکورد برای هر find, عملا رهاسازی با اتمام تراکنش (find دیگر), توصیه برای IX, اعمال قفل S, U یا X روی رکورد برای هر find, ارتقا در صورت نیاز به اصلاح, وگرنه مثل بالا برای SIX, عدم نیاز به قفل S روی رکورد برای find, لزوم قفل X برای اصلاح

فصل سوم: همزمانی, ۳-۱۱ قفل قصدی یا intent (ادامه)

- **خلاصه:** امکان انجام **Sfind**, **Ufind** و **Xfind** (در صورت وجود) روی قفلهای ضمنی گارانتی شده قبلی به شرح فوق, ویا امکان حفظ طولانی تر قفل نسبت به مدت گارانتی شده
- **تعریف مجدد پروتکل IL:** لزوم اعمال قفل I (IS, IX و SIX) قبل از اعمال هر نوع قفل روی یک شی, مثلا برای X روی رکورد اول اعمال IX یا SIX روی جدول
- **نکته:** وجود قفل X یا S در سطح بالاتر ← عدم نیاز به قفل خاص در سطح پایین تر
  قفل SIX در سطح بالاتر ← نیاز به قفل X در سطح پایین تر ولی نه S و U
- **ترتیب قوت قفلها:** ش ۳-۲۱, گراف ترتیب قوت قفلها, ارتباط با جدول ش ۳-۱۹
  L1 ضعیف تر از L2 اگر N در محلی از ستون L1 جدول ← N در همان محل از ستون L2 شکست یک تراکنش تحت L1 (ضعیف تر) ← شکست آن تحت L2 (قوی تر)
- **اصل سوم پیاده سازی:** امکان تبدیل درخواست یک قفل به سطح قوی تر توسط سیستم با رعایت امنیت کامل, نبود قفل S در سیستم و استفاده از X به جای آن.
  ش ۳-۲۲ نمایش قفلهای اولیه و ترتیب آنها

فصل سوم: همزمانی, ۳-۱۱ قفل قصدی یا intent (ادامه)

- **درعمل:** اشیای قابل قفل در قالب گراف بدون سیکل به جای عناصر منفرد,
  وجود اشیای وابسته مثل ایندکسها, زنجیره اشاره گرها, . . . برای هر جدول
  مثال ش ۳-۲۲, رجوع به یک رکورد ← لزوم رجوع به شی یا اشیای مادر
- **شرح کامل پروتکل IL:** برای هر تراکنش
  اعمال قفل X روی یک شی ← لزوم اعمال ضمنی قفل X روی فرزندان آن
  اعمال قفل S یا SIX روی یک شی ← لزوم اعمال ضمنی قفل S روی فرزندان آن
  قبل از اعمال قفل S یا IS روی یک شی ← لزوم اعمال قفل IS یا قویتر روی حداقل نسل قبل
  قبل از اعمال قفل IX, SIX, X یا U روی یک شی ← لزوم اعمال قفل IX یا قویتر روی همه مادرها
  قبل از لغو یک قفل از یک شی, لزوم لغو کلیه قفلهای اعمال شده روی فرزندان آن
- **مثال:** برای اجرای دستور UDL **find next (P where P.Color = 'red');**
  a) اعمال قفل IS یا IX روی SPDB, (IX برای PRLVL(CHG) و گرنه IS)
  b) اعمال حداقل قفل IS روی جدول P, (نوع دقیق وابسته به PRLVL و SHLVL طبق ش ۳-۲.)
  c) قفل روی P نوع IX, SIX یا X ← اعمال حداقل IX روی XP, (نوع دقیق مشابه قفل روی P)
  d) عدم وجود قفل S, X یا SIX روی P ← اعمال قفل S, U یا X روی رکورد یافت شده P

فصل سوم: همزمانی, ۳-۱۱ قفل قصدی یا intent (ادامه)

- خلاصه مشخصات قفلهای X, S, SIX, IX و IS (مهمترین موضوع این بخش)

وجود قفلهای S, X یا SIX توسط Tروی B ← عدم امکان اعمال تغییر تراکنش دیگر روی B

قفلهای X یا IX توسط Tروی B ← امکان اعمال تغییر T روی B (برخلاف S)

مزیت X نسبت به IX (روی جدول): عدم نیاز به قفل X روی رکوردهای جدول, کاربرد زمان لزوم اعمال اصلاحات زیاد روی جدول مثل زمان تجدید سازمان DB, اجتناب از ایجاد تعداد زیادی قفل, پرشدن فضای جداول قفل و شکست سیستم

قفلهای IS یا IX توسط Tروی B ← امکان اعمال تغییر تراکنش دیگر روی B

قفل IX توسط Tروی B ← امکان اعمال تغییر توسط T روی B (برخلاف IS)

- یادآوری نهایی: بی معنی بودن قفلهای SIX, IX و IS در پایین ترین سطح (رکورد)

فصل سوم: همزمانی, ۳-۱۲ سطوح جداسازی

- **موضوع:** حدود نیاز به اطلاع داشتن تراکنش عامل روی جدول B از اعمال تغییرات همزمان روی B (با فرض SHLVL(CHG) برای تراکنش)؟
- **پاسخ:** با توجه به سطح جداسازی تعریف شده برای B تحت تراکنش مزبور
  سطح جداسازی بالاتر ← تداخل کمتر در نتیجه همزمانی کمتر
- **پنج سطح ممکن:** ص ۱۱۹ و ۱۲۰, شمول سطوح بالاتر توسط سطوح پایین تر
  ۱: عدم اجازه این تراکنش برای بهنگام سازی تغییر CMT نشده در جدول تحت استفاده
  ۲: عدم اجازه این تراکنش برای دیدن تغییر CMT نشده در جدول مورد استفاده
  ۳: عدم اجازه هیچ تراکنشی برای تغییر رکوردهای داخل جدول, در اختیار این تراکنش
  ۴: عدم اجازه هیچ تراکنشی برای تغییر رکوردهای داخل جدول, که این تراکنش می بیند
  ۵: عدم اجازه آگاهی این تراکنش از سایر تراکنشهای مرتبط با جدول مورد استفاده
- **اصل چهارم پیاده سازی:** امکان تبدیل درخواست یک سطح جداسازی به سطح قوی تر (بالاتر) توسط سیستم با رعایت امنیت کامل
- **مثال:** جلوگیری از مشکل ش۳- ۲۴ درصورت وجود سطح ۳ جداسازی, سطح ۴ برای ش ۳-۷

فصل سوم: همزمانی، ۳-۱۲ سطوح جداسازی (ادامه)

- **فانتوم:** عدم امکان جلوگیری سطح ۴ از فانتوم، مثلا انتخاب رکوردهای با موجودی K۲.
عدم امکان جلوگیری از اصلاح به یا ایجاد چنین رکوردی، لزوم وجود سطح ۵ به شرح فوق،
لزوم قفل کردن مسیر دسترسی در سطح ۵ (عدم تغییر ایندکس مربوطه تا پایان کار)
مشابه قبل لزوم قفل کردن عدم وجود داده های خاص در DB
- **خلاصه:** قفل کردن رکوردها در سطوح فوق، لزوم وجود قفل IS یا IX روی جدول به دلیل SHRLVL(CHG)، پنج سطح ممکن
۱: UPD R ← قفل روی R توسط تراکنش دیگر منجر به تضاد، وگرنه اعمال قفل X روی R تا پایان تراکنش
۲: علاوه بر سطح ۱ بالا، Find R ← قفل X روی R توسط تراکنش دیگر منجر به تضاد، وگرنه ادامه کار
۳: علاوه بر سطح ۲ بالا، Find R ← اعمال قفل S روی R، رهاسازی با تغییر کرسور
۴: علاوه بر سطح ۳ بالا، حفظ قفل S تا پایان تراکنش
۵: علاوه بر سطح ۴ بالا، Find R ← اعمال قفل S روی مسیر دستیابی به R، حفظ تا آخر تراکنش
- **تکلیف ف ۳:** بررسی موضوع قفل روی محیطهای کتاب و حد اقل سه محیط دیگر

فصل سوم: همزمانی, ۳-۱۳ برچسب زمانی

- **روش:** تخصیص زمان شروع تراکنش به عنوان برچسب زمانی, اجرا بر مبنای برچسب,
عدم اعمال هر گونه اصلاح در صورت وجود برخورد, برگرداندن تراکنش و سپس اجرای مجدد با برچسب جدید به منظور ردیف شدن آنها, اعمال همه اصلاحات در زمان CMT
- **تفاوت:** در قفل, اجرای تراکنشهای متداخل به نحو ترتیب پذیر (هر ترتیبی)
در برچسب, اجرای تراکنشهای متداخل به نحو ترتیب پذیر (با یک ترتیب خاص)
- **حالات برخورد:** درخواست ترکنشی برای دیدن رکورد اصلاح شده توسط تراکنش جوانتر
درخواست تراکنشی برای اصلاح رکورد دیده شده یا اصلاح شده توسط تراکنش جوانتر
- **نتیجه:** عدم RLBK به دلیل Restart, عدم نیاز به قفل به دلیل اعمال اصلاح در CMT

مثال: نمونه های ص ۱۳۳