**جلسه اول:**

DBMS مانند قلعه برای پادشاه که همان داده هست، هست و دسترسی به داده سخت هست. یک سیستم مدیریت پایگاه داده هست و از داده محافظت میکند. داده و عملکرد حتما در هر محصول نرم افزاری هستند. در اولین قدم باید ارتباط بین مهندس و مشتری رو بچینیم. یعنی تهیه لیست نیازمندی ها و توسط ارتباط گر انجام میشود و دستاورد دارد که همون لیست نیازمندی هاست. ارتباط یعنی اینکه به مشتری بگیم بگو بینم چی میخوای. ارتباط نقطه صفر ما هست. در قدم بعدی باید تحلیل و طراحی بشود. به تحلیل و طراحی مدل سازی یا چک نویس میگوییم. آدم عاقل اول چرک نویس میکند و بعد باید برود سراغ پیاده سازی همون پاک نویسی هست. مدل سازی ساده شده یک واقعیت هست. تحلیل شبیه زبان انسان هست. ارتباط زبان انسان هست. طراحی به زبان شبیه ماشین هست. خود زبان ماشین میشود پیاده سازی. باید تحلیل را به طراحی تبدیل کنیم که میگویند نگاشت. طراحی الگو مناسبی هست برای پیاده سازی. خروجی هر فعالیت از اون 4 تا ورودی فعالیت بعدی هست. انتزاع یعنی کلی گویی یا حذف جزئیات. ما از یک فضای انتزاعی باید به یک فضای غیر انتزاعی برسیم. تحلیل انتزاعی تر هست یا طراحی؟ تحلیل هر چه قدر از بالا به پایین حرکت میکنیم انتزاع کم میشود. دقت کن هم داده هم عملکرد باید تجزیه و تحلیل و پیاده سازی شوند. واسه هر دو هست. با تحلیل موجودیت ظرف میسازند که ظرف همان جدول این ها هست و با پرس و جو بازیابی و ذخیره داده میکنیم. ابزار پرس و جو میشود حساب رابطه ای. مدل ER ابزاری هست برای تحلیل موجودیت. قبل از طراحی جدول باید تحلیل موجودیت انجام بدهی همین برای پرس و جو. بنابراین مثلا INSERT که با دیتا کار دارد پرس و جو و ساختن جدول طراحی جبری یا قسمت ظرف هست. نرمال سازی قبل پیاده سازی و بعد طراحی هست. نرمال سازی طراحی را تمیز میکند. هم طراحی جدول هم طراحی پرس و جو که همون جبر رابطه ای هست هم باید نرمال سازی شوند. سوال این است که بدانیم ما چند تا ظرف داریم به نوعی جدول داریم؟

بزرگ ترین مسئله این هست که ما چند تا ظرف داریم و پیدا کردن ظرف ها؟ بهترین راه حل این هست که ما اصلا چی را میخواهیم ذخیره کنیم؟ اگر مقدار تولید کننده داریم باید ظرف تولید کننده داشته باشیم. اگر مقدار قطعه داریم باید ظرف قطعه داشته باشیم. برای تحلیل موجودیت مدل ER میگه بگرد دنبال اسامی عام در محیط عملیاتی یعنی همونجا که اون برنامه قرار هست باشد. به اسامی عام همان موجودیت یا entity گفته میشود و باید با مستطیل نمایش داده شود. به این فکر کن چی را میخوای ذخیره کنی تا بفهمی موجودیت ها چی هستند این خیلی بهتر هست چون میخواهیم ذخیره کنیم پس باید موجودیت هم داشته باشیم. و موقعی که موجودیت ها را بدست آوردی باید صفت ها را بدست بیاری. حالا چطوری بفهمی صفت ها چی هستند؟ باید ببینی از یک موجودیت چی هارو دوست داری ذخیره کنی. بیضی نشانه صفت یا ویژگی هست. لوزی نشانه رابطه هست. رابطه بین دو موجودیت. ER با ERD فرق میکند ER یک سری قانون هست مثلا رابطه خواستی بکشی لوزی و ... ، بعد که کلا کشیدی به این میگویند ERD.

نوع رابطه چند به چند هست چون تولید کننده میتواند چند قطعه تولید کند. و از اونور هم قطعه توسط چند کارخانه تولید میشود. ظرف ها در تحلیل کاملا انتزاعی هست. مدل ER سه بخش مهم دارد رابطه موجودیت و صفت. خروجی مدل ER میشود ورودی طراحی جدول ابزار آن مدل رابطه ای هست. مدل رابطه ای همون ظرف ها هستند منتها باید به زبان ماشین بکشیم و ابزار مدل رابطه ای میگوید هر موجودیتی که اون بالا داری تبدیل کن به جدول. به صفت ها ستون های جدول گفته میشود. رابطه هم به شکل یک جدول میشود که نشان میدهد کدام قطعه را کدام شرکت به چه تعداد تولید کرده هست یک قطعه را میتواند دو شرکت تولید کند و برعکس. در اینجا غیر انتزاعی تر شدیم. قدم بعدی پیاده سازی هست. پیاده سازی هم دو بخش دارد. تو پیاده سازی پرس و جو ما با مقدار کار داریم و تو قسمت پیاده سازی جدول ظرف ها را میبینیم. پس ما باید طراحی جدول رو ببینیم برای پیاده سازی. یعنی شبه ماشین را ببین طراحی شبه ماشین هست تحلیل شبه انسان.

به دو زبان میشود حساب رابطه ای نوشت. این قسمت برای تحلیل هست. زبان تحلیل زبان نزدیک به انسان هست و زبان طراحی نزدیک به ماشین هست. جبر رابطه ای برای طراحی 3 قدم: 1. مکان جستجو یا همون جدول. 2. سطر جستجو 3. ستون جستجو. یعنی اول دنبال چیزی خواستی بگردی اول باید مکان جستجو رو مشخص کنی بعد بری سراغ سطر بعد بری سراغ ستون مورد نظر. سلکت تو SQL ستون را انتخاب میکند و تو جبر سطر را. به نحوی اون سطر یا ستون مثلا شهر فلان میشود شرط جستجو. شرط جستجو جایی هست که تیک و ضربدر باید بزنیم تیک جایی که باید انتخاب شود ضربدر جایی که نباید انتخاب شود. یعنی هر جا شرط برقرار بود تیک بزن برقرار نبود ضربدر بزن.

**فصل دوم بخش اول:**

فصل دوم هستیم راجب قسمت تحلیل جایی که به زبان انسان نزدیک هست قسمت ظرف. مدل ER برای فعالیت تحلیل هست برای تحلیل موجودیت یا ظرف جایی که شبیه به زبان انسان هست. برای طراحی جدول. در تحلیل منظور از رابطه ارتباط بین موجودیت ها هست و در طراحی منظور جداول طراحی رابطه میباشند. به موجودیت اسم عام هم گفته میشود این موجودیت در اون محیط عملیاتی هستند. به این فکر کن چی رو میخوای ذخیره کنی همون میشود موجودیت.

اسم خاص میشود رکورد یا تاپل یا نمونه موجودیت به ظرف میگویند اسم عام به محتوای اون ظرف میگویند اسم خاص. صفت چی هست و از کجا پیدا کنیم؟ خصوصیات یا ویژگی های یک موجودیت چون هر موجودیتی یک خصوصیت دارد. دقت کن چی رو دوست داری ذخیره کنی. هر صفت یک نام و دامنه دارد. قوانین مدل رابطه ای میگوید ستون های ما یک محدوده ای داشته باشد که مثلا جای عدد حرف وارد نکند. پس حتما صفت ها باید دامنه داشته باشند. تو تحلیل بگو صفت تو طراحی بگو ستون. انواع صفت: صفت کلید: از مقادیر جدول و محتوا نباید کلید پیدا کنی برای پیدا کردن کلید یا صفاتی که دارای خاصیت کلید هست باید عالم خارج را جستجو کنیم یعنی محیط زیست برنامه را باید جستجو بکنیم. برو ببین چه صفتی یا صفاتی هست که بتوانی بر اساس آن جستجو بکنی و یقه یک نفر را گرفت آورد اون کلید هست. قدرت تفکیک در حد 1 نفر باید باشد مثل کد ملی یا شماره دانشجویی. هر چیزی که یقه 2 نفر را گرفت آورد کلید نیست. دقت کن در محیط زیست خودش باید باشد. صفت کلید نباید null باشد. چون همه امید ما کلید هست. قانون جامعیت موجودیت میگوید کلید جدول نباید null باشد. در مدل ER هر صفتی که خاصیت کلیدی دارد یک خط میکشند. صفت ساده چیست صفت مرکب چیست؟ صفت ساده صفتی هست که مرکب نباشد. صفت ساده قابل تجزیه به اعضای با معنی نیست. مرکب قابل تجزیه به اجزای با معنی هست. ساده مثلا یک عدد هست دیگه چی رو میخوای تجزیه کنی. کد ملی هم ساده هست. تلفن هم ساده هست. مرکب: تاریخ تولد. یا آدرس. صفت ساده اتمیک هست و قابل تجزیه نیست. صفت مرکب را در مدل ER به صورت یک بیضی مرکزی و بیضی های کناری که به تکه های دیگر تجزیه شده اند هست. صفت تک مقداری: صفتی هست که یک نمونه موجودیت یا اسم خاص بیشتر از یکی نمیتواند داشته باشد و اگر چند تا داشت میشود چند مقداری مثلا کد ملی تک مقداری هست. صفت میتواند انواع ویژگی ها را داشته باشد. صفت کلید نباید مرکب و چند مقداری باشد در آینده توضیح میدهیم. چند مقداری با دو خط نشان میدهند. یا با 2 تا بیضی. صفت مشتق یا پویا: صفتی هست که میشود مقدارش را از صفت دیگری بدست آورد. ایستا همیشه ثابت هست پویا هعی عوض میشود هر ثانیه مثل سن. یعنی چیزی که از یک چیز دیگه بدست میاد را نباید در جدول گذاشت مثل سن از تاریخ تولد. صفت پویا مصیبت هست و جدول کثیف هست. چون هر لحظه در حال تغییر هست و وابسته به یک چیز دیگر هست بدرد نمیخورد. و بهتر هست نیاد. تو نمودار مینویسیم ولی در جدول نمی کشیم. ماهیت معدل یک صفت پویا هست که مداوم تغییر میکند. چون کل جدول باید تغییر کند.

**فصل دوم بخش دوم:**

رابطه: یک لوزی نام دار هست که بین موجودیت ها رابطه دارد. خواص رابطه : 3 خاصیت مهم دارد درجه ارتباط دومی کاردینالیتی ارتباط و سومی اجباری و اختیاری هست. همه روابط این خاصیت ها را دارند البته اگر رابطه ای باشد. همه روابط در صفات با هم اشتراک دارند ولی در مقدار صفات با هم متفاوت هستند. درجه ارتباط: مثلا ارتباط درجه 1 میشود یک موجودیت دانشجو نمونه های موجودیت رابطه همکلاسی دارند تعداد موجودیت های شرکت کننده در رابطه را میگویند درجه رابطه. که اگر 1 باشد میشود درجه 1. چه کسانی رابطه دارند؟ نمونه دانشجو با یک نمونه دانشجو دیگر در ظرف دانشجو و میتوانند رابطه همکلاسی داشته باشند. ارتباط درجه 2: دقت کن مدل ER خروجی میشود طراحی جدول. هر مدل ER نگاشت خاص خودش را دارد. چه درجه 1 چه درجه 2 چه چند به چند چه یک به چند. درجه 2 : 2 تا موجودیت مختلف شرکت کرده اند پس درجه 2 هست. دقت کن تحلیل بیشترین درجه را میگیرد. کاردینالیتی: میتواند یک به یک، یک به چند و چند به چند باشد. یک به یک: اگر هیچی نگوید پیش فرض اختیاری هست. در حالت یک به یک، یک به چند غلط هست. اگر نمونه موجودیتی شرکت نکرد باید ضربدر بزنی. رابطه اختیاری، اختیاری داریم رابطه اجباری هم داریم. کسی هم شرکت نکند تو رابطه چیزی گیرش نمیاد.

فلش دو طرف یک به یک ولی اگر یک طرف هیچی نبود یعنی یک به چند. در یک به یک رابطه اختیاری بود در یک به چند هم رابطه دو طرف اختیاری هست. دقت کن که کدوم را باید یک بگیری و کدام را چند. دقت کن از اون طرف خانم حداکثر با یک آقا هست. اگر هم شرکت نکند هیچی گیرش نمیاد.

اگر کلا فلش نداشت چند به چند هست. روابط قانون ندارند به محیط عملیاتی مربوط هستند یک جا استاد رابطه یک به چند دارد یا یک به یک یا چند به چند. بر اساس محیط عملیاتی هست. کاردینالیتی ارتباط یک خصوصیتی دارد به نام حد یا ارتباط یعنی حداقل و حداکثر را مشخص میکند. در رابطه یک به چند هر دانشجو یک استاد دارد ولی یک استاد چند دانشجو دارد البته دقت هم کن اختیاری هست یعنی استاد میتواند دانشجو نداشته باشد. موجودیت اختیاری: نمونه هاش هم اختیاری هست. اجباری: دو خط در طرفین رابطه میگذارند. همه روابط کاردینالیتی را دارند. کلا همه روابط اون 3 خصوصیت را دارند منتها در مقادیر متفاوت هست. دو خط را دیدی اجباری بودن اونیکه بهش وصل هست را مشخص میکند. حداقل صفر نشانه اختیاری هست. خط عمود مال اون یکی هست. یعنی اجباری هست کی اجباری هست؟ اون یکی طرف. در رابطه حضور نمونه ها همه اجباری هست. دایره یعنی اختیاری یعنی صفر. اون خط یک هست یعنی خط عمودی و اجباری یک طور متضاد هم هستند. این صفر هم همین شکل هست یعنی دایره اینوری یعنی اونوری اختیاری هست.

**فصل دوم بخش سوم:**

وجه تمایز موجودیت ها صفت های آن ها هست. درون موجودیت بین نمونه موجودیت وجه تمایز آنها، در یک موجودیت خاص میشود در مقادیر هست. وجه شباهت نمونه موجودیت ها در صفات هست. همیشه سعی کن از طرف اجباری بگیری که راحت تر هست. دقت کن کاردینالیتی نباید بهم بخورد.

این جلسه دوباره دیده شود.

**فصل سوم بخش اول:**

مدل رابطه ای: تو قسمت طراحی هستیم، حساب رابطه ای شبه کد نزدیک به زبان انسان و جبر رابطه ای شبه کد نزدیک به زبان ماشین هست الان ما تو طراحی جدول هستیم و مدل ER باید به جدول تبدیل کنیم صفت ها تبدیل به ستون میشود هر موجودیت یک جدول. مدل رابطه ای از بخش رابطه و قوانین بدست آمده است. قبلی 3 تا بود اینجا 2 تا تو قبلی رابطه منظور ارتباط بود اینجا جدول هست یعنی زیر مجموعه ای از ضرب دکارتی 2 تا دامنه تو هم. به قوانین جامعیت هم میگویند که از 2 بخش داخلی، خارجی دارد مثلا کلید نباید خالی باشد داخلی هست و خارجی اونایی هست که ما تعیین میکنیم مثلا مقدار حساب نباید زیر صفر باشد داخلی و خارجی را DBMS مدیریت میکند. مدل رابطه ای به زبان ماشین نزدیک هست و در سمت ظرف هستیم. سطح انتزاع دارد کم میشود یعنی از کلی به جزئی میرویم. حالا اول برویم رابطه: دامنه میشود به مجموع مقادیر یک صفت. با دستور create domain میتوانیم دامنه خودمان را شخصی بسازیم. به یک زیر مجموعه دلخواه از اون حاصل ضرب ها میگویند یک رابطه.

اون حاصل ضرب ها یک مجموعه هستند که نظم و ترتیب ندارد مجموعه عضو تکراری ندارد. در یک مجموعه اعضا میتوانند جا به جا شوند. رابطه هم صد در صد عضو تکراری ندارد. صد در صد رابطه هم عضو هاش میتواند جا به جا شود. به هر دو قسمت زوج مرتب میگویند یک عضو.

رابطه ها هم عضو تکراری ندارد و عضو ها میتوانند جا به جا شوند. تهی زیر مجموعه همه مجموعه هاست و هر رابطه میتواند شامل همه اعضای یک مجموع باشد که بهش میگوییم زیر مجموعه محض. به هر کدوم از این زوج مرتب ها یک تاپل گفته میشود. دقت کن یک رابطه مال ضرب دکارتی دو تا دامنه باشد یعنی همه اعضا و تاپل ها باید همگن باشند. جدول یک شیوه نمایش رابطه هست. یعنی میخواهیم پیاده سازی کنیم و از نظر کاربرد رابطه جدولی دارد. ببین ما یک صفتی دارید که دامنه آن را ما مشخص میکنیم یا همون دیتا تایپ. به هر سطر دیتا بیس میگویند سطر یا رکورد البته تاپل هم گفته میشود. خصیصه های یک موجودیت مثل شماره دانشجویی را که ستون میشوند میگوییم صفت خاصه ویژگی های اون ظرف هستند. پس صفت ها باید دامنه مقادیر داشته باشند. هر ستونی یک روزی صفت بوده هست. ببین همون مدل رو دیدی حاصل ضرب دکارتی دو تا دامنه؟ همون میشود جدول دقیقا یعنی مدل کامپیوتری همون هست. و اگر اولی عضو دامنه 1 بود بقیه هم باید عضو دامنه 1 باشند عضو های اول همین برای عضو دوم اون تاپل.

**فصل سوم بخش دوم:**

صفت به ستون تبدیل میشود. جدول نوع نمایش دیگری از رابطه هست. زوج مرتب ها را میتوانی جا به جا کنی. ترتیب صفت های خاصه در یک تاپل مهم نیست یعنی میشود s را با c تغییر داد البته به شرطی که ترتیب صفت خاصه در تمام تاپل های دیگر همین شکلی باشند و عوض شده باشند. همه موارد صفت خاصه نباید تجزیه پذیر باشند. یعنی اینکه نباید از تاپل های تو در تو یا داده های مرکب باشند مثل تاریخ تولد. دقت کن تاپل در تاپل اصلا نمیتوانیم تولید کنیم و حضور آن غلط هست. چون یک مجموعه تاپل در تاپل ندارد پس زیر مجموعه هاش یا رابطه هاش هم ندارد و کلا نداریم و باید اصلاح کنیم. و باید نرمال کنیم که یعنی مثل تاریخ تولد بازش کنیم. مدل رابطه از دو بخش رابطه یا همون جدول و قوانین تشکیل شده است که دیدی ما رابطه را معرفی کردیم اینجا خواص رابطه رو بررسی میکنیم: درجه رابطه و کاردینالیتی رابطه. درجه رابطه میشود تعداد ستون ها و کاردینالیتی رابطه میشود تعداد سطر ها مثلا اونجا میگفتیم رابطه یک به یک یا یک به چند هست اینجا میشود تعداد ستون ها. هر جدولی درجه و کاردینالیتی دارند اما در مقادیر متفاوت هستند عین قبلی. انواع کلید: ابر کلید، کلید کاندید، کلید اصلی، کلید فرعی، کلید خارجی. دقت کن هنوز وارد قوانین نشدیم فعلا فقط رابطه و راجب جدول صحبت میکنیم. ابر کلید: یا super key، یا رجال سیاسی. کلید فرعی میشود کلید کاندید منهای کلید اصلی. کلید اصلی کاملا سلیقه ای هست. کلید اصلی کاندید هست. کلید اصلی هم ابر کلید هست. ابر کلید: خاصیت کلیدی داشته باشد. هر چیزی که بر اساس جستجو برود یقه کسی را بگیرد بیاورد میشود ابر کلید. همه خاصیت کلیدی دارند دقت کن کلید خارجی جز این کلید ها نیست و فعلا دقت نکن بهش. اصلا کلید را نباید بر اساس محتوا جدول انتخاب کرد باید از عالم بیرون باشد. ترکیب کلید جدول مثلا با یک نام دیگر چون یقه یک نفر را میگیرد و میاورد پس خاصیت ابر کلید را دارد. یعنی ابر کلید میتواند چند ستونی باشد فقط تک ستونی نیست. چرا رجال سیاسی نتوانست کاندید شود؟ چون حاشیه داشت پس برای رئیس جمهور شدن اول باید رجال سیاسی باشی بعد صلاحیت هم تایید شود و حاشیه نداشتی باشی. رجال سیاسی ابر کلید هست. چیز میز اضافه همون ستون های اضافی هستند یعنی وقتی یک کلید به تنهایی یقه یک نفر را میگیرد و میاورد دیگر لازم نیست چیز میز اضافه باشد. شرط لازم این هست که کلید باشد شرط کافی این هست تنها باشد و چیز میز اضافه باشد. کلید کاندید نمیشوی اگر این 2 شرط را نداشتی. پس اگر خاصیت کلیدی نداشته باشد دیگر اصلا کلید نیست که بخواهد کلید کاندید باشد یا نباشد. دقت کن میتواند یک چیز ابر کلید باشد و دو ستونی باشد و بعد کلید کاندید بشود. یک جدول n کلید ابر کلید میتواند داشته باشد. کلید هم n تا یا مثلا بی حاشیه ها منهای با حاشیه ها. کلید فرعی: کاندید منهای اصلی میتواند فرعی نداشته باشد.

**فصل سوم بخش سوم:**

کلید خارجی: بهتر بود اسم ارتباط دهنده میان دو جدول را براش نام گذاری میکردیم. کلید الزام بر خاصیت کلیدی داشتن ندارد یعنی نمیرود یقه یک نفر را بیاورد. پس بنابراین میتواند مقادیر تکراری داشته باشد. یکی از علت های وجود کلید خارجی بحث نرمال سازی هست. نرمال سازی کار تمیز کاری هست مال جدول هست. دقت کن کوئری ها هم باید نرمال سازی کنیم. مقادیر درون جداول از عالم خارج اومدند. تعریف ساختاری جداول فرعی یا پل: کلید های خارجی میتوانند ترکیبی باشد از کلید دو یا چند جدول دیگر و این کلید ها در جدول خودش شرط خاصی ندارد یعنی الزاما خاصیت کلیدی ندارد و دوم اینکه در همان یا جدول دیگری خاصیت کلیدی داشته باشد. در جدول خودش منظور خاصیت درجه ارتباط 1 هست. جداول دیگری میشود درجه ارتباط 2 و 3 و4. دقت کن این صفت ها در جدول خودش شرط خاصی ندارد ولی در جدول دیگری باید کلید کاندید یعنی یا اصلی یا فرعی باشد علت هم بحث نرمال سازی هست. به چه دردی میخورد؟ برو تو اون جدول فرعی یک رکورد مثل s1 را بردار بعد ببر تو جدول مربوط به خودش که اصلی هست بری اونجا تو جدول خودش میشود s1. ارتباطات به ما قدرت میدهد. پس نگاه کن در جدول خودش شرط خاصی ندارد کلید خارجی ولی در جدول اصلی قدرت میگیرد زیرا اونجا کلید کاندید هست. این از دیدگاه ساختار. دیدگاه محتوایی: دقت کن برای برقراری کلید خارجی هر دو تعریف ساختاری و محتوایی درست باشد. محتوایی میگوید که مقادیر کلید خارجی در جدول فرعی زیر مجموعه محض مقادیر کلید کاندید در جدول اصلی باشند. انگار جدول اصلی آسمان هست و جدول فرعی زمین هست. چرا زیر مجموعه محض هست؟ چون اون کلید رو میگیریم میبریم داخل جدول خودش میبینیم روی هیچ کسی نمیفتد بنابراین قانون محتوایی یا دیدگاه محتوای نقض میشود یعنی کسی بی صاحب نماند. یعنی هر مقدار باید متناظر در جدول اصلی داشته باشد. کلید خارجی نباید هیچ موقع ارجاع null باشد چون قانون جامعیت ارجاعی نقض میشود. خواستی کلید خارجی پیدا کنی اول برو تو ستون مشترک رو پیدا کن البته میتونی نگاه هم کنی مقدار تکراری هم میتواند داشته باشد این شد از لحاظ ساختاری از لحاظ محتوایی باید بری ببینی همه سطر ها متناظر باشند در سطرهای جدول اصلی. ببین کلید خارجی میتواند مقدار null داشته باشد البته به شرطی که در شرایطی که در قوانین بالا دستی قرار نگیرد. قوانین مدل رابطه ای: جامعیت ارجاعی: کلید خارجی نباید دچار ارجاع null شود یعنی نباید به null اشاره کند. جامعیت موجودیت: میگوید که نباید کل کلید کاندید یعنی هم اصلی هم فرعی null بشود. یعنی اون کلید خارجی چون داخل تعریف کلید کاندید هست نباید کل کلید یا بخشی از اون null بشود. دقت کن از لحاظ ساختاری ستون مشترک باشد و اون کلید باید جز کلید کاندید باشد. دقت کن چون کلید خارجی میفتد داخل کلید کاندید هم نمیتواند تکراری باشد چون داخل اون جدول جز کلید کاندید هست هم نمیتواند null باشد چون جز قوانین بالا دستی هست چون اونجا کلید کاندید هست.

نگاشت: چندین نوع نگاشت داریم مثلا اگر درجه 1 باشد به یک شکلی نگاشت میشود تازه همون درجه 1، باید کاردینالیتی مشخصی هم داشته باشد مثلا یک به یک تازه باید مشخص شود که اجباری یا اختیاری هست. همیشه وقتی موجودیتی در ارتباط یا پل نیاز داریم که کاردینالیتی چند به چند باشد. چون معنی نمیدهد قطعه تعدادش را خود قطعه حمل کند باید تولید کننده حمل کند پس نیاز به یک پل داریم. مدل رابطه ای گفته است که از خروجی مدل ER که میشود ERD استفاده کن و نگاشت کن. مدل رابطه ای میگوید را تبدیل کن به جدول ظرف بشود جدول صفت بشود ستون. ببین تو پل ببین چه وابستگی هایی داریم بین کلید ها اون رو بکن کاندید. مدل درجه 2 از نظر کاردینالیتی سه مدل دارد بعد 2 دسته هم میشود اختیاری و اجباری ولی توی مثالی که مطرح شده فرق نمیکند اجباری یا اختیاری یعنی هر چی بود هست.

**جلسه سوم بخش چهارم:**

چون رابطه یک به یک هست ذاتا نمیشود ترکیبی باشد یا تکراری باشد. کلید خارجی: آقا راحت هست بگیرش ببر اونور ببین یقه یک نفر را میگیرد بیاورد یا نه بهت قدرت میدهد یا نه. برای اختیاری اجباری کلید اختیاری را ببر به عنوان یک سطر در جدول اجباری ها بگذار. به جای جدول پل. اگر در رابطه اختیاری بودند مقدار آنها NULL میشود. کلا ببین هر نوع جدولی میشود طراحی کرد بحث ما بهینگی است. فرض را بر بهینگی بگذار.

دو تا اجباری درون یک جدول میتواند باشد. در یک به چند یک مدل نگاشت هست و برای اجباری اختیاری فرقی نمیکند دو تا جدول داریم دقت کن قبلی ها همه واسه نگاشت های 1 به 1 بودند. در یک به چند یک را ببر داخل جدول چند بگذار.

**جلسه سوم بخش پنجم:**

دقت کن رابطه رو نمونه موجودیت ها ایجاد میکنن. مثلا وقتی رابطه درجه 2 هست نمونه های مثلا کارمند با نمونه های اداره ارتباط برقرار میکنند. در 1 به 1 نمونه های مثلا دانشجو با نمونه های همون دانشجو ارتباط برقرار میکنند. در یک به یک باید یک جدول دیگر اضافه کنی که کلید خارجی میشود کلید کاندید و اصلی جدول یک به یک و دقت کن کلید خارجی حتما نباید با کلید اصلی نام یکسان داشته باشد. کلا شرط این هست که تعداد ستون ها یکی باشد و مقادیر دامنه یکی باشد. دقت کن چرا در جدول دومی که میسازیم خود s# رو نگرفتیم؟ چون خود اینها دانشجو ها هستند اون sp# هم کلاسی های اینها هستند. دقت کن کلید خارجی رو اول برای بحث محتوایی مطرح میکنند مثلا یک دانشجو میخواهد با یک دانشجو دیگر ارتباط داشته باشد در صورتی که تو جدول اول نمیشود اضافه کرد راهکار این هست که کلید خارجی تعریف میکنیم و هم ارتباط برقرار میکنیم و هم تضمین میدهیم که مقادیر کلید خارجی زیر مجموعه کلید اصلی باشد پس کی نمیگذارد که s5 وارد شود؟ دیدگاه محتوایی نمیگذارد هم پروژه ای دانشجو سوم شود. دقت کن واسه جدولی که مثلا به نام هم پروژه ای میسازیم دقت کن که هر دو کلید باید زیر مجموعه جدول اصلی باشند. این واسه اختیاری بود. حالا بریم سراغ اجباری دقت کن کلید خارجی در جدول خودش الزامی به یکتا بودن ندارد ولی در جای دیگر باید یکتا باشد. برای اجباری باید یک جدول داشته باشیم. دقت کن در این حالت میشود کلید خارجی و کلید اصلی در یک جدول باشند. دقت کن ترجیح ما این هست که تک جدولی رفتار بکنیم. حالت بعدی همان درجه ارتباط یک هست ولی کاردینالیتی ارتباط یک به چند هست یعنی بعضی از کارمندان هستند که چندین کارمند دارند در موضوع اختیاری داریم بررسی میکنیم. در واقع یک نمونه با نمونه های دیگرش رابطه بک یه چند دارد. برای این بخش هم یک جدول میکشیم. و مثلا دقت کن هر شخص حداکثر یک رئیس دارد و یک رئیس چندین کارمند دارد یک به چند هست دیگر. اینجا اصلا حالت اجباری اختیاری اصلا نداریم چون کلا یک موجودیت داریم یا کلا اجباری هست یا اختیاری مثلا در همین قسمت رئیس اختیاری هست دیگه چون s1 رئیس ندارد پس تو رابطه شرکت نکرده است پس اختیاری هست. در این مثال کارمند و رئیس اجباری بودن معنی ندارد. پس همیشه رابطه یک به چند به شکل تک جدولی هست همیشه.

کلید خارجی باید از نظر محتوایی زیر مجموعه کلید کاندید باشد.

رابطه چند به چند در نگاشت رابطه 1 به 1: براش کشیدن درخت معنی نمیدهد. تک جدولی هم نمیشود. حتما باید با 2 جدول بکشیم. چرا تو اون مثال جفت گفتید کلید خارجی؟ چونکه باید حتما زیر مجموعه کلید کاندید در جدول اصلی باشند و از آن تجاوز کند.

**جلسه سوم بخش ششم:**

رابطه وراثت یا ISA، رابطه وراثت یک رابطه پدر و فرزندی هست. هم پدر با فرزند رابطه دارد هم فرزند با پدر. دقت کن هر فرزند یک پدر دارد ولی پدر میتواند چند فرزند داشته باشد. پدر میتواند اجباری یا اختیاری با بچه ها در ارتباط باشد. چیزی که پوشا هست غیر مجزا هست یعنی با هم پوشانی دارند و disjoin یا غیر پوشا یا مجزا هم نام دارد. دقت کن شکل هم بیضی با یک دایره داخل میتواند باشد برای شکل overlap. دو صورت داریم واسه هر کدوم از این روابط فرزند واسه disjoin میشود یک بیضی یا یک D داخلش. رابطه اجباری و اختیاری واسه پدر هست و رابطه disjoin overlap مال فرزندان هست. دقت کن فعلا تو فاز تحلیل هستیم و مدل EER هست. این چیزها رو بخاطر این گفتم که نمیشود پدر همزمان هم فرزند overlap داشته باشد هم فرزند disjoin. یعنی از هم مجزا هستند. یعنی مثلا نگاه کن ما یک ماشین داریم نمیشود 2 تا پلاک داشته باشد. این میشود مجزا یا غیر پوشا. حالا پوشا مثال میشود اینکه تو یک شماره مشتری در یک بانک داری ولی هم میتوانی حساب قرض الحسنه داشته باشی هم حساب جاری یعنی پدر از هر دو فرزند میتواند همزمان داشته باشد ولی در مجزا نمیتواند. در disjoin حد فرزند (1,1) هست یعنی حداقل و حداکثر باید یک فرزند داشته باشد چون اجباری هست که برود تو رابطه. دقت کن در disjoin کلا یک فرزند تحت هر شرایطی که باشد یعنی تکرار صورت نمیگیرد فقط یک فرزند و داخل آن هم تکرار نباشد. همیشه رابطه فرزند با پدر اجباری هست ولی علائم را در تحلیل نمیگذارند به صورت پیش فرض هست. در پوشا یا overlap تکراری وجود دارد شماره مشتری میتواند 100 ها حساب جاری داشته باشد. پس disjoin فقط یکی تکرار نداریم.

نگاشت: به صورت بهینه 2 جدولی هست، برای disjoin و دقت کن باید تمام صفت های پدر را کپی کنی در جدول فرزند یعنی باید باباش رو بیاری جلو چشمش، دقت کن disjoin و در صورت اجباری بودن. دقت کن تمام پدر را باید بیاری داخل فرزند. مدل سه جدولی هم هست که کلید پدر را میاوریم در جدول های پایین ولی بهینه نیست چون سربار حافظه دارد. همیشه تحت هر شرایطی فرزند نسبت به والد اجباری هست.

در رابطه یک به چند گفتیم یک را ببر بگذار در چند، در چند به چند گفتیم 3 جدول که یکی شامل صفت رابط این 2 هست. یک را گذاشتی در چند میشود به عنوان کلید خروجی یادت باشد.

**جلسه سوم بخش هفتم:**

پوشا و اختیاری را بررسی میکنیم. بچه ها از پایین به بالا اجباری هستند پدر میتواند اجباری یا اختیاری باشد. کلید پدر را فقط بیار پایین در جدول ها و به عنوان کلید خارجی درج کن. دقت کن 3 جدولی میشود اختیاری اجباری فرقی نمیکند. اینجا پایین به بالا حتما اجباری هست ولی دقت کن پدر میتواند فرزندی نداشته باشد مثلا با این شماره مشتری حسابی نداشته باشیم. چی باعث شد با این شماره مشتری هم اینور باشد هم اونور؟ overlap کی باعث تکرار شد؟ overlap. طراحی 2 جدولی غیر بهینه هست یا افزونگی دارد چرا؟ مثلا هعی فامیلی ها تکرار میشود هر دو طرف آدرس هعی تکرار میشود ولی در صورتی که با یک کلید خارجی شماره مشتری اصلا نیازی به این همه تکرار نداشتیم. مقادیر null هم زیاد دارد چون اختیاری هست. معمولا همیشه overlap با اختیاری میاد و disjoin با اجباری میاد و سایر ترکیب ها را نداریم.

موجودیت ضعیف: وابسته حتما به صورت اجباری باید در رابطه باشد و یکی گیرش بیاد. مثلا بیمه از طریق پدر که دانشگاه فرزند را نمیشناسد. اون موجودیت که این بهش وابسته هست میتواند اختیاری باشد به اون اصلی دید سر پناه داشته باش. به اون شماره وابسته صفت ممیز گفته میشود و به شماره سر پناه شماره شناسه گفته میشود. حالا در طراحی اولا حتما باید 2 جدول باشد. شماره وابسته میتواند تکراری باشد در بیرون خاصیت یکتایی ندارد ولی در داخل خاصیت یکتایی دارد این شماره وابسته. موجودیت ضعیف با 2 مستطیل تو در تو نمایش داده میشود همچنین کلید کاندید آن در جدول 2 تایی هست همچنین شماره وابسته حتما دارد و حتما اجباری هست و حتما صفت ممیز دارد. به کلید کاندید در سر پناه صفت شناسه هم گفته میشود و دقت کن شماره وابسته داخل یکتا و بیرون یکتا نیست. خط چین نشانه صفت ممیز و خط کامل نشانه کلید یا صفت شناسه هست.

قوانین مدل رابطه ای: 2 تا دارد داخلی خارجی، خارجی همان قوانین حکومتی هست که نمیتوانی از آن تخطی کنی خارجی را ما تعریف میکنیم ولی داخلی را خود DBMS طراحی کرده است. مثلا ما طراحی کردیم نمره 0 تا 20 باشد کی اجرا میکند و نظارت میکند؟ DBMS، داخلی را مدل رابطه ای تعریف میکند. درون رابطه ای میگوید درونت باید آدم باشد جدول باید جدول باشد اگر جدول، جدول نباشد جدول نیست حالا درون رابطه ای میگوید صفت چند مقدار و مرکب نباید داشته باشد و تاپل تو تاپل شده است، باید حداقل یک کلید کاندید داشته باشد حالا چطوری درست میشوند در نرمال سازی باید تمیز کنیم. کلید تکراری باشد درون رابطه ای رد میکند. قانون جامعیت موجودیت: همون اینکه کلید نباید null باشد و قانون جامعیت موجودیت نقض شده است و نباید بخشی یا همه کلید کاندید null باشد. فرعی باید از اصلی تبعیت بکند ارجاع null به این شکل هست که مثلا اگر یک کلید خارجی باشد که اون در جدول اصلی نباشد ارجاع null را نقض میکند به این میگوید جامعیت ارجاعی. قانون دامنه ای: محتوایی که وارد میکنی با data type ستون یکسان باشد.

**فصل چهارم بخش اول:**

جبر رابطه ای: جبر مربوط به فعالیت طراحی هست که باهاش جستجو میسازند. مدل شبه جدول هست جبر شبه کد هست. اینها شبه پیاده سازی هستند. راجب عملوند های جبر رابطه ای صحبت کردیم که همون جداول یا مدل رابطه ای هستند و باید راجب عملگرها صحبت کنیم.

|  |
| --- |
| عملگرهای فرعی موقع استفاده از اصلی ها کمک میگیرند یک جورایی فرعی ها خلاصه شده اصلی ها هست و از خودشان قدرتی ندارند و از اصلی قدرت میگیرند به نوعی انتزاعی تر شده اصلی ها هستند مثلا برای اشتراک به جای اینکه بگوییم 2 تا تفاضل میگوییم اشتراک. در پرس و جو نویسی: اول باید مکان جستجو مشخص باشد، دقت کن هر چیزی یاد میگیریم در SQL هم همین شکلی هست. دوم باید شرط جستجو را بیان کنیم که به سطر اشاره دارد و سوم خروجی جستجو یا ستون مورد نظر ما هست. یعنی جدول سطر ستون یا R,select,project. عملگر select: دقت کن تمام توضیحات یکسان هست برای هر عملگر، این عملگر بدرد جستجو میخورد دقت کن که ما راجب مکان جستجو در مدل رابطه ای کاملا صحبت کردیم. |

این عملگر select همیشه رو یک جدول کار میکند پس تک عملوندی هست. R مکان جستجو و تتا شرط جستجو هست و خروجی به ما میدهد. درجه رابطه مقصد: یعنی تعداد ستون ها، راجب ورودی ها صحبت نمیکنیم چون تو مدل رابطه ای صحبت کردیم راجب خروجی این دستورات صحبت میکنیم. اگر محدودیتی برای خروجی جستجو ندارد یعنی هر چی ستون هست بیاد تو خروجی. وقتی select میزاری یعنی همه سطر ها نه و وقتی project میزاری یعنی همه ستون ها نه، اگر نگذاشتی یعنی همه سطر ها یا ستون ها میتوانند بیان عملگر سلکت وظیفه سطر ها را دارد تیک و ضربدر هست یعنی کیا باید بیان کیا نباید بیان. پس همیشه خروجی عملگر سلکت درجه جدول مبدا با درجه جدول مقصد یکی میشود چون اصلا به ستون کار ندارد عملگر سلکت و فقط به سطر کار دارد. کاردینالیتی رابطه: بهش تعداد سطر ها گفته میشود و همیشه بزرگتر مساوی صفر هست. درجه همیشه بزرگتر از صفر هست. کاردینالیتی مقصد همیشه کوچکتر از کاردینالیتی مبدا هست دقت کن همه اینها راجب سلکت هست دقت کن. کاردینالیتی اگر همه سطر ها بیاد برابر و اگر چند سطر بیاد کوچکتر از مبدا هست ولی درجه مبدا و مقصد یکی هست چون هیچوقت دستکاری نمیشود. نکته:

تا دقیقه 40 دیده شد.

ما توسط عملگر سلکت میتوانیم 2^n یا ترکیب صفر از 3 تا 3 از 3، جدول درست کنیم. کلید کاندید سلکت: دقت کن سلکت روی درجه تاثیر میگذارد ولی دقت کن محتوا تاثیری تو انتخاب کلید ندارد و ستون تغییر نمیکند در عملیات سلکت، و مجموعه انتخاب شده زیر مجموعه جدول اولی هست. ما در جبر رابطه زیر مجموعه بودن ستونی نداریم یعنی مثلا نمیتوانیم اون چیزی که زیر مجموعه هست یک سری ستون نداشته باشد بر اساس تاپل هست. زیر مجموعه بودن ما همیشه سطری هست، شرط اول این هست که تعداد ستون ها باید یکسان باشد و همچنین باید از نوع و دامنه ستون های مورد نظر هم یکسان باشند به طور کلی هم تیتر باشند در این موقع هست که میتوانیم راجب زیر مجموعه بودن نظر بدهیم، پس وقتی این زیر مجموعه اون هست کلید هر چیزی که مجموعه اصلی کلید اون جدول زیر مجموعه هم هست. دقت کن پرس و جو از درونی ترین پرانتز بررسی میشود. دقت کن خروجی های شرط ها با هم and میشوند. یاد آوری چون به ستون دست نمیزند درجه ها یکی هست. کلید ها یکی هست ولی کاردینالیتی یکی نیست چون به سطر دست میزند.

عملگر project: پرتو یا تصویر نیز میگویند، البته به سلکت هم انتخاب گفته میشود. خود عملگر شبیه ستون هست پس ستون انتخاب میکند، تتا سطر انتخاب میکرد چون شرط بود و L ستون انتخاب میکند. درجه رابطه کوچکتر مساوی جدول مبدا هست چون رو ستون ها دستکاری میکنی مثلا مساوی هم میتواند باشد چون همه ستون ها را انتخاب کند. کاردینالیتی مقصد کوچکتر مساوی مبدا هست. دقت کن حالت کوچکتر کاردینالیتی این شکلی هست ما یک ستون را برمیداریم ولی چون اون ستون مقادیر تکراری دارد و تعداد سطر ها کمتر میشود این حالت اتفاق میفتد. با سلکت گفتیم 2^N حالت حالا برای پرتو میگوییم 2^N-1 حالا چرا؟ چون نمیتوانیم هیچ ستونی را انتخاب نکنیم. کلید کاندید: سلکت چون به ستون دست نمیزند پس مبدا و مقصد یکی ولی پرتو چون به ستون دست میزند پس کلید مبدا و مقصد یکی نیست کلید پرتو دو فرم دارد، (هیچوقت کلید به محتوا کاری ندارد)، فرم اول این هست که ستون کلید را حفظ کند پس کلید همون هست، دقت کن اثبات زیر مجموعه را نمیخواهیم داشته باشیم و هم تیتر باشن یا نباشند ربطی ندارد فقط همین ستون کلید بود و به طور کامل اومد نتیجه کلید همون هست. فرم 2: چون ستون کلید نیامده، به صورت تمام کلید میشود یعنی همه ستون ها کلید هستند دیگه کاری به تکراری و محتوا نداشته باش. جدولی که تمام کلید میشود فقط یک کلید کاندید دارد. قاعده کارکرد تو در تو: باید ستون های بیرونی زیر مجموعه ستون های داخلی باشند. چون اول باید پهن تر باشد بعد هعی تنگ تر میکنیم. خاصیت جا به جایی: پرتو ها نمیتوانند جا به جا باشند چون جا به جا کنیم قاعده تو در تو نقض میشود بعضی اوقات درست نیست فقط در حالتی درست هست که قاعده تو در تو نقض نشود یعنی L1=L2 باشد دقیقا. مقایسه جا به جایی پرتو و سلکت: یعنی جای این 2 را عوض کنیم مثلا یکی را بیاوریم بیرون همیشه درست نیست همیشه هم غلط نیست به شرطی درست هست که سلکت زیر مجموعه پرتو باشد مثلا تتا زیر مجموعه L باشد. مثلا تو ستون الف را با پرتو میاری ولی تتا ستون الف و ب را میخواهد چطوری میتوانی همچین کاری کنی نمیشود.

**فصل چهارم بخش دوم:**

وقتی همه ستون ها را میخواد دیگر لازم نیست پرتو بزاری تو جدول میانی باید اول یک شرکتی یک قطعه ای تولید کند بعد بیاد تو این جدول میانی. اجتماع: اجتماع جدولی با هر جدولی نمیشود بلکه باید شروط سازگاری داشته باشد یا هم تیتر باشد یعنی هم درجه یکسان هم نوع و دامنه یکسان. از لحاظ کاردینالیتی: درجه که میدانی یکسان هست. دو نوع کاردینالیتی داریم بیشترین حالت که میشود جمع دو تای قبلی به نوعی که اشتراک تهی باشد بیشترین مقدار میشود. در کمترین مقدار این هست که اشتراک دارند و ببین کی تو دل کی هست و اجتماع ما میشود اون گنده که کمترین مقدار ممکن هست. دقت کن در جبر رابطه ای تکرار نداریم و تاپل تکراری نداریم مثلا در اجتماع فقط یکبار ظاهر میشود. اجتماع هر رابطه با خودش میشود خودش. فرمول اجتماع(همه جا جواب میدهد): نه سقف دارد نه کف یه چیزی اون وسط هست، k سطح های مشترک هست. کلید کاندید اجتماع: ( دقت کن همه عملگر ها کلید مخصوص دارند) کلید کاندید اجتماع همیشه تمام کلید هست. ببین دقت کن کلا محتوا رو بیخیال نشو بر اساس محتوا تصمیم نگیر ولی یک نمونه محتوا بیار ببینیم درست هست یا نه.

در جدول زیر کلید اصلی خط میکشند اگر زیر هر 3 ستون خط کشیدند یعنی تمام کلید هست، اجتماع خاصیت شرکت پذیری و جا به جایی دارند و خاصیت توزیع پذیری هم دارند چه مثلا 2 تا اجتماع بگیر بعد سلکت کن یا جدا به جدا شرط اعمال کن سلکت کن بعد اجتماع بگیر مثلا. در اجتماع تکراری ها حذف میشوند. اشتراک: هم تیتر بودن هم برای اشتراک هم برای اشتراک هم برای تفاضل هست. درجه پس باید برابر باشد، حداقل اشتراک میشود تهی از نظر کاردینالیتی یعنی صفر میشود چون عدد هست، از نظر حداکثر وقتی یکی تو دل یکی دیگه هست منتها اون کوچک تر دیگه طبیعتا. و به طور میانگین میشود k، در اجتماع اشتراک آنها را کم میکردیم واسه حد متوسط ولی در اینجا دقیقا برابر خود اشتراک هست. کلید کاندید اشتراک: یا کلید اصلی جدول اول یا کلید اصلی جدول دوم جفتش نمیشود چون میشود ابر کلید. کلید سلکت همیشه برابر همون کلید مبدا هست چون به ستون کاری نداشت. برای زیر مجموعه بودن اول باید هم تیتر باشند و شرط دوم این هست که کل مقادیر دومی در دل اولی باشد یعنی کوچک در دل بزرگ باشد. اشتراک به نوعی برای جداول نقش سلکت را اعمال میکند یعنی جدول اول باید کلش بیاد در مقصد ولی جدول دوم نمیگذارد انگار اون شرط هست. اشتراک خاصیت جا به جایی و شرکت پذیری و توزیع پذیری دارد. این توزیع پذیری چه به ازای جدول یکسان و شرط متفاوت و به ازای جدول متفاوت یا شرط یکسان یکی هست. عملگر تفاضل: شرط انجام پذیری هم تیتر بودن هست. در تفاضل جدول مقصد صد در صد زیر مجموعه جدول اول است. و جدول دوم نقش سلکت را برای جدول اول بازی میکند. کلید آن هم کلید جدول مبدا هست. از نظر درجه که یکی هست. از نظر کاردینالیتی سقف میشود زمانی که اشتراک نداشته باشند و کاردینالیتی برابر میشود ولی کف میشود کاردینالیتی صفر یعنی جدول پایه درون اون جدول دوم که مانند سلکت عمل میکند باشد. حالت متوسط هم میشود کاردینالیتی جدول پایه منهای اشتراک.

**فصل چهارم بخش سوم:**

در تفاضل خاصیت جا به جایی نداریم، شرکت پذیری هم نداریم، در توزیع پذیری شرط جدول دوم که نقش سلکت را بازی میکرد میتوانیم حذف کنیم، چون اون شرط ها را انگاری در جدول پایه اعمال کردیم، شبیه سازی: فرعی ها را میتوان از روی اصلی ها ساخت همه فرعی ها داخل آنها اصلی دارند انتزاعی تر هستند میتوانی جزئیات را حذف کنی و سبب بالاترین رفتن خوانایی میشود. پرس و جو نویسی: وقتی گفتند یک چیزایی شامل یک چیزایی نباشد یعنی تفاضل. عملگر ضرب دکارتی: هر وقت بخواهیم همه سطر های یک جدول را کنار همه سطر های یک جدول دیگه قرار بدهیم. کاردینالیتی برابر هست با حاصل ضرب کاردینالیتی 2 جدول مبدا، کلید را باید به صورت 2 تایی بگیری که اون کلید کاندید ما میشود کلید اصلی جداول مبدا، درجه میشود مجموع درجات 2 جدول مبدا. ضرب دکارتی کاری با ستون مشترک ندارد. یعنی هر ترکیبی از کلید های جدول پایه دقت هم کن ابر کلید نیست، ابر کلید باید چیز اضافه داشته باشد. خاصیت جا به جایی دارد، شرکت پذیری هم هست.

در پرس و جو دوم همه سطر ها بدرد ما نمیخورد چون پیوند پذیر نیستند، پیوند نا پذیر ها اطلاعات الکی هستند اون شرط را میگذاری که پیوند نا پذیر ها حذف شوند. نرمال سازی فقط واسه جدول ها نیست واسه پرس و جو هم هست و دقت کن نرمال سازی قبل از پیاده سازی هست، ضرب دکارتی قدرت گرفتن هست با کسی که هم تراز خودمون هست و پیوند پذیر هستیم. اگر s را sp ضرب کنیم پیوند پذیر معنی دارد ولی اگر s را در p ضرب دکارتی کنیم اصلا معنی ندارد چون ستون مشترک نداریم، ضرب دکارتی حتی وقتی دنبال همه حالت های ممکن هم بودی خوب هست. پس اگر گفتند جفت هایی را بگو که مثلا قطعه تولید نکرده اند یا آزمون شرکت نکرده اند اول بیا همه دانشجو ها را ضربدر آزمون ها بکن بعد منها کن از اون دانشجو هایی که آزمون ثبت نام کرده اند و ستون مشترک هستند.

**فصل چهارم بخش چهارم:**

عملگر الحاق طبیعی: بعضی از عملگر ها یک سری شرط خاص دارند مثل این عملگر که برای اینکه الحاق انجام دهد نیاز دارد به ستون مشترک اگر ستون مشترک نداشت همون ضرب دکارتی هست، اگر ستون مشترک داشت سطر های پیوند پذیر را میدهد مثلا دکارتی شرط خاصی ندارد ولی اجتماع و اشتراک و تفاضل باید هم تیتر باشند. یک فرق دیگر این هست که ستون مشترک فقط و فقط یکبار میاد بر خلاف دکارتی که ستون مشترک 2 بار تکرار میشد. خروجی هم سطر های مشترک هست. درجه میشود جمع درجه 2 جدول منهای اشتراک آنها. خود جدول مجموعه هست و مجموعه ای از تاپل ها هست. دقت کن در پرس و جو نویسی میتوانند از عملگر اجتماع استفاده کنند جای شرط چون دقیقا همین حکم شبیه به این عملگر را دارد. دقت کن ستون مشترک میتواند 2 تا باشد اصلا ربطی ندارد. کاردینالیتی هم میشود کوچکتر مساوی حاصل ضرب کاردینالیتی 2 جدول اول که ستون مشترک داشتند. دقت کن میتواند تهی هم باشد هر گاه که ستون های مشترک مقادیر عددی کاملا مشابه نداشته باشند. این حرف غلط هست که هر موقع دنبال الحاق عملگر طبیعی هستیم دنبال کلید خارجی باشیم الزامی ندارد و غلط هست و فقط باید ستون مشترک باشد. ولی اگر در الحاق عملگر طبیعی استفاده شود و مفهوم کلید خارجی تعریف شده باشد کاردینالیتی جدول به وجود آمده از کاردینالیتی جدول دوم که در آن کلید خارجی تعریف شده است کمتر مساوی است. دقیقا میشود همون جدولی که کلید خارجی داشته است. یک چیزی کلید خارجی میتواند مقدار null داشته باشد ولی نباید ارجاع null داشته باشد. کلید کاندید: وقتی میتوانیم نظر بدهیم که کلید خارجی داشته باشیم اگر داشتیم کلید کاندید میشود کلید کاندید جدولی که در آن کلید خارجی تعریف شده است. خاصیت جا به جایی و شرکت پذیری دارد. اگر کلید خارجی نداشتیم میشود کلید 2 جدول قبلی به صورت یک کلید. اگر دو جدول کاملا هم تیتر باشند الحاق عملگر طبیعی میشود همان اشتراک این 2 جدول. برای بیشترین حالت یا ستون مشترک داشته باشد و مقدار ها یکی باشد یا ستون مشترک کلا نداشته باشد که میشود ضرب دکارتی. زیر پراجکت میتوانی اجتماع یا اشتراک بگذاری چون از جنس مجموعه هستند. عملگر نیم پیوند: 2 نوع داریم چپ و راست اونور که بسته هست مثلا چپ پس میشود نیم پیوند چپ، هر دو جدولی این عملگر ها برقرار نیست، باید شرایط عملگر الحاق طبیعی برقرار باشد، اگر ستون مشترک نداشتیم عملگر نیم پیوند تبدیل میشود به ضربی که جلوتر توضیح میدهیم. اول اصلا فکر کن عملگر الحاق طبیعی هست خروجی را بدست بیار فقط دو تا اول ستون چپ را بردار انگار جدول چپی رو میخوای از خروجی بردار اینطوری فکر کن که جدول دوم برای جدول اول نقش سلکت را بازی میکند در نیم پیوند چپ. پس از لحاظ کاردینالیتی کوچکتر مساوی جدول اولی هست. کلید آن میشود کلید جدول اول چون زیر مجموعه همون جدول اول هست. درجه هم دقیقا با جدول اول برابر هست. نقش سلکت یعنی اینکه اون سطر در خروجی برود یا نه. بیشترین حالت کاردینالیتی این هست که ستون مشترک تماما مقدار یکسان داشته باشند یا مشابه باشند. کلید کاندید هم هر چیزی که r1 کلیدش هست دقیقا همان هم برای r3 که حاصل هست، هست. جا به جا پذیر نیست، جا به جا پذیری یعنی عملگر را حفظ کنیم عملوند جا به جا کنم دقت کن نیم پیوند راست و چپ با هم، هم ارز هستند. در موارد خاص که دو جدول اشتراک داشتند نیم پیوند چپ با الحاق طبیعی با اشتراک یکسان هست. اگر در نیم پیوند ستون مشترک نداشتیم در r2 و نتواند نقش سلکت را برای r1 اجرا کند میشود خود r1.

**فصل چهارم بخش پنجم:**

عملگر الحاق خارجی چپ یا فرا پیوند چپ: یک سری چیز های خارج از الحاق طبیعی را دارد یعنی از خارج از الحاق میاورد اضافه میکند. از خط های موازی بفهم چپ هست یا راست موازی چپ بود چپ و برعکس. کامل هم داریم که هم چپ را دارد هم راست را. اگر چپ بود یعنی الحاق بیشتری از چپ میخواهد. الحاق طبیعی شرط ستون مشترک را داشت و پیوند پذیر ها را میداد ولی فرا پیوند اون پیوند ناپذیر ها را هم بیار. و مقابل آنرا از جدول r2 null بزار. درجه آن دقیقا مثل الحاق طبیعی هست بحث بیشتر از الحاق در سطر ها بحث میکنیم در ستون که فرقی ندارد. پس همون حرفایی که واسه الحاق طبیعی میزنیم واسه فرا پیوند هم میزنیم.

کاردینالیتی: بزرگتر مساوی الحاق هست کی بزرگتر هست؟ پیوند ناپذیر داشته باشد کی مساوی هست؟ تماما پیوند پذیر باشد. و کوچکتر مساوی ضرب حسابی این 2 جدول r1 r2 هست. اگر r1 تهی بود الحاق تهی میشود دقت کن تو مورد قبلی پیوند ناپذیر بودن به این معنی نیست جدول تهی شود. چه وقتی میگوییم یک چیز زیر مجموعه چیز دیگری هست؟ باید هم تیتر باشند. کلا عملگر های مجموعه ای باید هم تیتر باشند. شرط لازم هست. شرط کافی این هست که آنچه اینور هست اونور هم باشد یعنی داخل اون یکی باشد. میتواند مساوی هم باشد مثلا پیوند ناپذیر باشد همه سطر ها یا مثلا تهی باشد جدول اول. هم تیتر بودن: از نظر درجه و نوع مثل هم باشند یعنی شروط سازگاری. نیم پیوند هم دقت کن جدول دوم نقش سلکت دارد و مشترک ها میاد تو جدول نهایی. دقت کن نیم پیوند چپ زیر مجموعه الحاق طبیعی نیست. کلید کاندید فرا پیوند چپ: نمیتوانیم راجب کلید نظر بدهیم چون اگر 2 تایی بگیریم قانون جامعیت موجودیت که میگوید تمام یا بخشی از کلید نباید null باشد زیر سوال رفته است همه کلید دارند جز اینها. جا به جا پذیر نیست. اگر ستون کاملا مشترک بود یعنی هم تیتر بودند فرا الحاق چپ تبدیل میشود به اشتراک و در همین شرایط کلید r3 برابر با کلید r1 میشود. یک مورد خاص دیگر این هست که ستون مشترک نداشته باشیم و تبدیل میشود به ضرب دکارتی. الحاق خارجی کامل یا فرا پیوند کامل: هم چپ را دارد هم راست را این هم بیشتر از یک الحاق هست اینطوری به خاطر بسپار فرقی هم نمیکند اول چپ را اضافه کنیم یا راست را. کاردینالیتی آن کمتر از ضرب دکارتی هست. دقت کن الحاق طبیعی زیر مجموعه فرا الحاق کامل هست همچنین همین رابطه برای فرا الحاق چپ و راست. در اجتماع باید 2 طرف هم تیتر باشند. اجتماع فرا الحاق چپ با راست میشود فرا الحاق کامل. اشتراک فرا الحاق چپ و راست میشود الحاق طبیعی. اجتماع فرا الحاق چپ و راست و کامل میشود الحاق طبیعی. راجب کلید کاندید هم نمیتوان نظر داد. جا به جا پذیر هست. فرا الحاق کامل همان اجتماع هست یا متناظر هستند جا همین اگر الحاق بزاریم میشود اشتراک البته اگر هم تیتر بودند. الحاق گرفتی انگار اشتراک گرفتی پس اول الحاق بگیر یا اشتراک بگیر بعد برو سراغ ناپذیر ها.

**فصل چهارم بخش ششم:**

وقتی گفتیم همه یعنی تقسیم شرط لازم هست نه کافی. برخی یا همه یا بعضی یا اولین و آخرین معنی ندارد چون اگر شرط برقرار باشد همه را میدهد دیگه این ادبیات را نداریم. هر وقت بررسی یک اتفاق یا رویداد را بررسی کنیم که همه آن را انجام داده باشند. زیر مجموعه ستون های مقسوم علیه باید زیر مجموعه ستون های مقسوم باشد. مقسوم علیه نقش سلکت را بازی میکند و خارج قسمت که همان a هست میاد در خروجی و این همان a در مقسوم هست هم میتوان خودش کامل بیاد هم گاهی اوقات بعضی ها بیان. یعنی میخواهیم بدانیم همه اون اتفاقات که در مقسوم علیه هست در مقسوم اتفاق افتاده هست که بیاد در خارج از قسمت یا نه. همه حالات باید اتفاق بیفتد یعنی 1 درست هست با چیز های دیگری آمده ولی چون با 3 و 4 هم آمده است پس مشکلی ندارد و در خارج از قسمت میاد. درجه میشود درجه اون 2 تا منهای همدیگر. پس ستون خارج از قسمت میشود ستون های مقسوم منهای ستون های مقسوم علیه یعنی مشترک ها را بنداز بیرون. از لحاظ کاردینالیتی حداقل صفر و حداکثر کاردینالیتی اولی تقسیم بر دومی هست طبیعی هم هست یا هیچی نمیاد یا هر چی که هست اتفاق افتاده. کلید کاندید: اگر کلید مقسوم به طور کامل رفت تو خارج از قسمت که هیچی کلید همان هست و تغییری نمیکند ولی اگر کلید مقسوم خراب شد در تقسیم باید تمام کلید بگیریم. واسه تقسیم یک چیز یادت باشد همه اتفاق های حالت های مقسوم علیه باید اتفاق افتاده باشد تا بیاد. یعنی اگر 56 با 12 اومد باید با 34 هم اومده باشد. خاصیت جا به جایی ندارد طبیعی هست. r میشود یک جدول R میشود ستون های آن. R-S یعنی ستون های مقسوم منهای ستون های مقسوم علیه یا خارج از قسمت. در منها هم تیتری باید برقرار باشد. به خارج از قسمت میگوییم کامل برای اون ردیفی که همه حالات براش اتفاق افتاده باشد برعکس همین برای ناقص کیا میرن تو خارج از قسمت؟ کامل ها. ناقص ها را چطوری پیدا کنم؟ از همون جدول r بیا a را بردار ضرب دکارتی کن در مقسوم علیه تا در جدولی به وجود بیاد که کل حالت های کامل به وجود بیاد. شبیه سازی در جبر را یاد بگیر یعنی حالات کامل ( که از طریق ضرب دکارتی بدست آمده) را منهای حالات موجود کن بعد پراجکت بگیر بعد سلکت کن. مقسوم علیه همه حالات با هم هستند اتفاق و مقسوم هم با هم هستند. دقت کن برای تقسیم باید زیر مجموعه باشد یعنی ستون های مقسوم علیه همه در مقسوم باشند. واسه همین پراجکت گرفتیم.

**فصل هشتم بخش اول:**

وابستگی تابعی: دو بخش دارد کامل یا FFD و وابستگی تابعی غیر کامل که در فصل نرمال سازی صحبت خواهیم کرد که خود این به چند طبقه تقسیم میشود وابستگی بخشی و انتقالی و معکوس و چند مقداری و الحاقی. وقتی اشتراک دو جدول را نوشتیم اون تمام کلیدی که حاصل از دو کلید اصلی هر کدام از این 2 جدول هست را بهش کلید اصلی میگوییم که هیچ منتها هر جز از این کلید اصلی را به تنهایی عضو کلید یا صفت عمده گفته میشود و یک فرقی با غیر کلید دارد. چون چیزی که بخشی از کلید هست را بخشی از کلید گفته میشود. دقت کن ابر کلید یک تعریف تئوری هست یعنی در جداول چیزی به نام ابر کلید نداریم یعنی نمیگن این جدول ابر کلیدش چیست پس عضو کلید بخشی از کلید کاندید هست بخشی از ابر کلید نیست. وابستگی تابعی هر موقع از ابر کلید به یک غیر کلید رفتیم یک وابستگی تابعی یا FD هست. ابر کلید یک مفهوم تئوری هست عضو کلید بخشی از کلید کاندید هست. هر چیزی کلید کاندید باشد ابر کلید هم هست. غیر کلید بیاد غیر کلید را تولید کند FD نیست بهش وابستگی انتقالی در غیر کامل ها میگوییم. FD یعنی ابر کلید غیر کلید را تولید کند. اگر عضو کلید غیر کلید را بدهد بهش میگوییم وابستگی بخشی که در غیر کامل ها میخوانیم. این مواردی که تا الان خواندیم FD بود ولی الان میخواهیم FFD بخوانیم که در سمت چپ باید کلید کاندید باشد ( بر خلاف قبلی که ابر کلید بود) و سمت راست هم غیر کلید باشد. دقت کن ترکیب یک کلید با یک غیر کلید باز هم همان FD هست چون ابر کلید هست دیگه. FFD حتما سمت چپ کلید کاندید و راست غیر کلید باشد. دقت کن وابستگی یعنی جستجو کنی یقه یک نفر را بیاورد. کاندید، کاندید را بدهد بدیهی هست و FD, FFD نیست. بدیهی دیدی حذف کن.

چیزی که FD نیست FFD هم صد در صد نیست شرط لازم هست ولی هر چیزی که FD هست لزوما FFD نیست. چون ممکن است ابر کلید باشد و کلید عضو زائد داشته باشد و کلید کاندید نباشد. چه با ابر کلید چه با کاندید یقه یک نفر را میگیرد میاورد واسه همین وابستگی دارد. پس اگر در سمت چپ ابر کلید یا کاندید باشد سمت راست فقط یک خروجی دارد بر خلاف غیر کامل ها. وابستگی های بدیهی را باید حذف کنیم. اگر کوچک زیر مجموعه بزرگ باشد بدیهی هست دقت کن یعنی x->y پس باید YcX باشد حالا میخواد کلید باشد یا نباشد بدیهی ها را نباید حساب کرد. اینطوری فکر کن که سمت راستی زیر مجموعه سمت چپی باشد. دقت کن در قوانین آرمسترانگ دومی بدیهی نیست. با این 3 تا تمام روابط را میشد پیدا کرد بعد یک سری فرعی ها اومد. **قانون بستار**: یک جدول داریم که یک سری وابستگی دارد. دقت کن A را از خودش داریم طبق قاعده خودش، خودش را میدهد پس هر چی سمت چپ بود سریع بنویس تو خروجی دقت کن تو قاعده هر کدام خودشان را میدهند یعنی وقتی میگوییم ad-f یعنی هم a را داریم هم d را. مهم ترین کار در قوانین چهار گانه ارسطو نحوه کشف کلید کاندید هست. قانون اول تعیین سطح هست که ببینیم تهی میشود یا غیر تهی و به سمت قانون دوم و سوم. قانون اول همان قانون تفاضل هست: اجرای قانون اول ارسطو به دو مزه یا دو الگوریتم اجرا میشود روش اول یا دوم. روش اول باید خصیصه های کلی جدول را منهای غیر بدیهی ها کنی که البته روش دوم بهتر هست. اگر تهی شد که هیچ چون کلا 2 خروجی داریم یا تهی یا غیر تهی. اگر غیر تهی شد باید تعیین سطح کنیم تا بفهمیم قانون دوم هست یا سوم. اگر در بستار به همه رسیدیم میشود قانون دوم و اگر به غیر همه رسیدیم میشود قانون سوم. دقت کن وقتی چیزی جز قانون دوم شد یعنی فقط یک کلید دارد دیگر چک نکن برای کلید کاندید. دقت کن از قانون اول تفاضل وقتی استفاده میکنیم که غیر بدیهی باشد در غیر اینصورت استفاده نمیکنیم یعنی منها کنیم سمت راستی هایی که غیر بدیهی هستند را فقط منها میکنیم.

**فصل هشتم بخش دوم:**

پس مسیر کلی به این شکل هست که منها میکنی حاصل را بستار میزنی ببینی به همه میتوان رسید یا نه اگر بشه رسید قانون دوم هست و کلید مشخص هست. دقت کن فقط قانون دوم یک کلید کاندید دارد. راه حل تستی: منها کردی توش بود اون گزینه هایی که توش این باقی مانده قرار نداره حذف کن چون عضو هیئت کلید هست و عضو کلید کاندید هم هست منتها میتواند کلید های دیگری هم باشند. اما سایر کلید ها چی؟ میتواند شامل سایر ستون ها هم باشد یعنی شامل بقیه خصوصیت ها البته همه را ندارد میایم ببینیم خود اون عضو کلید فعلی چی ها را میدهد؟ دقت کن قبلش باید خود اون خصیصه که عضو کلید هست به همراه هر چیزی که میدهد از اون بستار را خط بزنی بعد بین بقیه بگردیم حالا: به اسم روش بازگشتی هست برای کشف بقیه کلید ها یعنی ببین کدوم ها اون خصیصه را میدهند اون ها هم جز کلید کاندید هستند. بعد بستار میگیری باید همه را بدهند. حالا خود اون ترکیب اولی از کجا اومده؟ دونه به دونه تست کردیم ببینیم همه را میدهد یا نه از طریق بستار. دقت کن اونکه حاصل از منها میمونه به طور مشترک با سایر کلید ها وجود دارد. دقت هم کن اونایی که خود اون حاصل از منها میمونه بهش دسترسی داشت خط بزن. قانون سوم همین رو میگه که اون عضوی که مونده به عنوان عضو دائم در تمام کلید های کاندید هست. غیر بدیهی چطور میفهمیم؟ هیچکدوم زیر مجموعه چپی نیستند.

وقتی تهی شد یعنی وجه اشتراک همه تهی هستند و در کلید عضو مشترکی وجود ندارد برخلاف قانون سوم که کلی کلید داریم ولی وجه مشترک همه حضور اون باقی مانده از تفاضل هست. راه حل تستی واسه تهی این هست که از گزینه ها بستار بگیر هر کدام همه رو نداد نیست. اگر یکی از کلید های کاندید لو برود خوب هست چون بقیه رو از روی اون به شکل بازگشتی میسازیم. ببین دقت کن وقتی یکی همه را میدهد پس اونایی که این را هم میدهند هم کلید کاندید میشوند. ببین بین دو کلید ممکن هست یک جز کلید مشترک باشد ولی ما بین همه کلید ها منظورمان هست اشتراک تهی هست. دقت کن اونایی که جز بستار عضو کلید هستند و خط میزنیم طبق قانون سوم دیگر به هیچ شکلی عضو کلید کاندید نیستند. حتی اگر قانون دوم هم باشد جز نیستند چون فقط 1 کلید کاندید داریم. 2 جدول زمانی کاملا همانند هم هستند یا هم ارز هستند که 2 شرط برقرار باشد 1. هم تیتر باشند یعنی هم درجه و از نوع یکسان باشند. 2. وابستگی های دو جدول کاملا یکسان باشد به عبارت دیگر کلید دو جدول یکسان باشد. کلید را چطوری بدست میاورند؟ از روی وابستگی تابعی. پس اول ببین هم تیتر هستند بعد ببین وابستگی های یکسان دارد یا نه. یک روش دیگر این هست که بیایم وابستگی ها را یکسان کنیم تا کلید یکسان داشته باشند که توش میخواهیم یک چیز زیاد را کم کنیم از لحاظ وابستگی چون یکی اضافه هستند نیازی به حضور آنها نیست فرعی ها وابستگی هایی هستند که از روی اصلی ها قابل ساخت هستند پس نوشتن آنها معنی ندارد یک روش دیگر برای اصلی فرعی این هست که ببین اون وابستگی را حذف کن بعد ببین میتوانیم مثلا از A برسی به B یا نه اگر توانستی فرعی بوده هست که توانستی برسی اگر نتوانستی پس اصلی بوده است. دیاگرام حالت هم میتوانی بکشی. و اگر بخواهند دو جدول یکی بشوند باید اضافی ها حذف بشوند. در مرحله اول باید سمت راست ها تجزیه بشوند و تمام سمت چپی ها کوتاه شود و سومی فرعی ها حذف شوند. قانون یک و دوم یعنی دو تا صفتی ها یک صفتی شوند. وقتی یک چیزی یک چیز را بدهد دیگر نباید بیاد کنارش یعنی اگر A->B پس AB->D دیگر معنی ندارد B آنجا باشد. این 3 تا نکته برای کوتاه کردن هست.

**فصل نهم بخش اول:**

در این فصل میخواهیم نرمال سازی در جداول را پیاده سازی کنیم و پس از طراحی و قبل از پیاده سازی هست. نا سازگاری یا بی نظمی ( در اینجا فقط بهش میگوییم) این هست که مثلا شماره 12 اشاره میکند به دانشکده کامپیوتر ولی ما وقتی این شماره 12 را به 13 تغییر میدهیم باید همه این شماره ها در سطر ها تغییر کند ولی وقتی ما یادمان برود بعضی سطر ها را تغییر بدهیم سطر های حاصل دچار بی نظمی یا ناسازگاری شده اند و علت اصلی همه این موارد تکرار هست. ریشه سالم که همان وابستگی تابعی کامل یا FFD هست باعث میوه سالم یا جدول نرمال میشود. ولی اگر وابستگی تابعی ما غیر کامل باشد باعث جدول غیر نرمال میشود. و ریشه را باید از انواع وابستگی ها مثل معکوس یا انتقالی و ... پاک کنیم تا فقط وابستگی تابعی کامل برای ما باقی بماند آیا میشود یک جدول وابستگی تابعی کامل نداشته باشد؟ نه نمیشود چون به کلید کاندید ربط دارد و نمیشود یک جدول کلید کاندید نداشته باشد. 5 مشکل داریم چون 5 نوع تکرار داریم دقت کن اگر وابستگی بخشی را حذف کنیم درخت سالم نمیشود زیرا هنوز وابستگی های تابعی غیر کامل دیگری هم مانده اند.

پس حکایت نرمال حذف علت هست. مثلا با 2NF وابستگی بخشی را حذف میکنند که بهش فرم دوم نرمال گفته میشود که درمان مرض هست برای انتخاب میشود 3NF و برای بقیه هم همین شکلی به ترتیب و به همین ترتیب شروع میکنیم به درمان مرض. جداول نرمال شده باید بین آنها کلید خارجی تعریف شود که بتوانند به هم متصل شوند. میوه سالم منظور عدم تکرار و میوه خراب یعنی تکرار هست. کلید کاندید باید بهش ارجاع بشود. افزونگی های طبیعی یا محتوایی را وابستگی های غیر کامل پدید میاورند. افزونگی فنی یا تکنیکی را هم کلید خارجی یا شاخص تولید میکند و دقت هم کن بین افزونگی وابستگی تابعی کامل جایی ندارد چون اصلا افزونگی ایجاد نمیکند. 12 و 12 افزونگی فنی هست و ما فقط افزونگی محتوایی را حذف میکنیم و با حذف این افزونگی فنی شکل میگیرد تکنیکی قابل تحمل تر هست چون تکرار نداریم. دقت کن پس از نرمال سازی افزونگی حذف نمیشود. ولی ممکن هست بعضی از افزونگی ها حذف شوند طبیعی ها مطابق یک فرآیندی به ترتیب حذف میشوند ولی فنی ها به وجود میان. دقت کن در اون مثال بعدی که نیاز به الحاق ندارد حجم هر رکورد را به ما داده اند. دقت کن بعد از نرمال سازی موردی ندارد برای هر پرس و جو عمل الحاق بنویسیم، بستگی به نوع پرس و جو دارد. رکورد های قبل نرمال سازی بزرگتر هست ولی بعد از آن چون بخشی و کوچکتر میشوند زمان هم کمتر میشوند دقت کن همیشگی نیست یعنی هر موقع نرمال سازی کردیم زمان پرس و جو کمتر نمیشود چون وقتی نیاز به الحاق باشد کند تر میشود.

دقت کن بعد از نرمال سازی همه افزونگی حذف نمیشود بلکه افزونگی های جدید رخ میدهد. یعنی پرس و جو لزوما سریعتر نمیشوند و بستگی به پرس و جو دارد چون نرمال شده یک گام عقب تر هست در کل. موقعی که نیاز به الحاق داریم باید جدول بزرگ را از اول بسازیم یعنی وقتی الحاق نداریم آماده هست ولی وقتی الحاق داریم جدول آماده نیست و وقت گیر هست چون اول باید ضرب کنیم بعد تازه حساب و کتاب کنیم یعنی اگر بدون الحاق 12 باشد نیاز به الحاق x+12 هست که اون x هزینه اون ضربی هست که تا جدول را بسازیم واسه همین هست که لزوما سریعتر نمیشود پرس و جو های ما. ضرب هم دقت کن اول ضرب دکارتی میکنیم بعد پیوند پذیر ها را جدا میکنیم. نرمال سازی یک فرآیند هست و یهو اتفاق نمیفتد. اولین نرمالی که باید اتفاق بیفتد 1NF هست که ربطی به وابستگی ها ندارد بعد دقت کن هعی به درون حرکت میکنیم از بیرون که به این میگویند فرآیند نرمال سازی و درونی زیر مجموعه بیرونی هست (خودش گفته بیرونی زیر مجموعه درونی هست) یعنی جدولی که 1NF نباشد نمیتوانی 2NFش بکنی. یعنی اگر بخواهد چیزی 5NF باشد باید اول 1NF تا 4NF باشد بعد اعمال کنیم. اگر BCNF باشد یعنی 3NF بوده هست ولی جدولی که 2NF باشد نمیتوانیم نتیجه بگیریم 3NF هست یعنی وقتی تا یک سطحی رسیدیم نسبت به سطوح قبلی شایستگی داریم یعنی حائز شرایط پشت سرمان هستیم ولی حائز جلوی سرمان نیستیم عین مدرک تحصیلی که کسی که ارشد دارد لیسانس هم دارد ولی برعکس درست نیست چرا از بیرون میرویم به داخل؟ خوب اینطوری میگیم که آدم هایی که دکتر هستند بیشتر هستند یا دیپلم؟ خوب مشخص هست دکتر پس خیلی از جداول ممکن هست 1NF باشند ولی 5NF نباشند. فرم اول نرمال: که بهش 1NF گفته میشود دقت کن که این فرم نرمال ارتباطی به وابستگی ندارد. شرط لازم و کافی دارد. نرمال سازی رویه کاهش هست کاهش چی؟ تکرار، تکرار چطوری از بین میرود؟ با حذف وابستگی بخشی و انتقال یعنی غیر کامل ها. رویه کاهش یعنی حذف افزونگی محتوایی. شرط لازم 1NF یعنی دارای حداقل یک کلید کاندید باشد یا همان جامعیت درون رابطه ای و شرط کافی این هست که ستون های چند مقداری و مرکب نداشته باشد که این هم جامعیت درون رابطه ای هست یعنی تاپل تو تاپل نباشد یا رکورد تو رکورد یا جدول تو جدول نباشد یعنی جدول باید جدول باشد جدولی که جدول نباشد جدول نیست. مثل تاریخ تولید که یک مثال مرکب هست و انگار یک جدول داخل جدول هست چون خودش فرم جدول دارد یا تاپل تو تاپل هست و مثال جدول غیر نرمال هست چون درون رابطه ای ندارد پس اگر مدل رابطه ای داشت یعنی فرم اول نرمال را لحاظ کرده ایم. جامعیت کلا دو بخش دارد خارجی و داخلی که خارجی توسط طراحان صورت میگیرد و داخلی که همان مدل رابطه ای هست که به 4 بخش درون رابطه ای ارجاعی موجودیت و دامنه ای تقسیم میشود که مهم ترین آنها همین درون رابطه ای هست و ما برای اینکه یک جدول را حساب کنیم باید اول بگیم فرم اول نرمال که همین قاعده درون رابطه ای هست را باید داشته باشد پس برای اینکه به فرم اول نرمال در بیاد باید قانون درون رابطه ای را پیروی کند پس باید فرم جدول تو جدول را اصلاح کنیم و باید 5 تا دامنه را در هم ضرب کنیم تا فرم نرمال اول ایجاد شود یا به اصطلاح درون رابطه ای را لحاظ کند و کلید هم که دارد پس صلاحیت سطح اول را دارد. مثلا در مثال اول بدون تجزیه جدول پایه و بدون تعریف کلید کاندید و صرفا فقط با درج ستون توانستیم به سطح اول نرمال برسیم دقت کن که این روش درج ستون بدرد چند مقداری هایی میخورد که محدود باشد مثلا راجب شماره تلفن جواب نمیدهد چون یکی ممکن است 10 تا شماره تلفن داشته باشد و هعی باید جدول درج کنیم در صورتی که یکی هم 1 موبایل دارد و 9 تا مقدار NULL دارد. روش دوم که برای نامحدود ها هست، این هست که 2 تا جدول کنیم بعد با کلید خارجی دوری و دوستی را لحاظ کنیم تا دوباره بتوانیم با الحاق به یک جدول واحد برسیم و بدون کلید خارجی نمیتوانیم نرمال سازی کنیم چون دوری و دوستی نمیشود. بعد از نرمال سازی باید جداول را تنها در نظر بگیری که ببینی فرم نرمال سطح 1 هست. تو 1NF گاهی اوقات تجزیه میکنیم گاهی اوقات به 2 جدول تبدیل میکنیم چه محدود چه نامحدود باشد. در مرکب هیچ وقت تجزیه نداریم فقط تو چند مقداری تجزیه داریم مرکب با درج جدول هست.

**فصل نهم بخش دوم:**

علت افزونگی طبیعی در جداول چیست؟ وابستگی های افزونه ساز یا همان 5 علت همون خراب کار ها که تکرار درست میکنند هر کدام به شکل خودشان نا مطلوب چیست؟ کلید خارجی بین آنها تعریف نشود مثلا یا ارتباطی نباشد. دقت کن هر جوری خواستی نمیتوانی نرمال سازی کنی، اگر هر جوری خواستی نرمال سازی کنی میشود نا مطلوب مثلا جوری تجزیه کنی که ستون مشترک نداشته باشد شرط لازم برای کلید خارجی ستون یا ستون های مشترک هست که الزامی بر هم نام بودنشان نیست البته از فرم نرمال دوم به بعد چون تو اولی یک جا هایی تجزیه میخواست یک جاهایی میخواست یعنی تو مرکب تجزیه نمیخواد ستون اضافه میکردیم تو چند مقداری هم اگر کم بود نمیخواد و ستون اضافه میکنیم، اگر زیاد بود به اجبار تجزیه میکنیم. تجزبه مطلوب دو بخش دارد که در شرط لازم آن در قسمت بی کاست گفته میشود که نه چیزی به جدول اضافه شود و نه چیزی از جدول کم شود یعنی دقت کن به صرف داشتن ستون مشترک حل نیست باید به کلید خارجی تبدیل شود برای رسیدن به این شرط لازم که بی کاست و بی حشو برسیم باید به کلید خارجی برسیم اولی یعنی چیزی کم نشود و دومی یعنی چیزی اضافه نشود که خود این با تعریف کلید خارجی بدست میاد. شرط لازم میگوید باید حافظ FD باشیم. در 2NF باید وابستگی بخشی را حذف کنیم. و به کمک تجزیه مطلوب نرمال سازی میکنیم که شرط لازم تجزبه مطلوب بی کاست بی حشو هست و شرط کافی هم حافظ FD باشد. پس مراحل به این شکل هست که اول جدول با وابستگی را میدهند ما قوانین ارسطو را میزنیم و کلید جدول را پیدا میکنیم و با استفاده از وابستگی میبینیم که وابستگی بخشی داریم که عضو کلید کاندید غیر کلید را میدهد که در این مثال هم دقیقا همین هست.

دقت کن که قبل از تجزیه باید کلید خارجی را تعریف کنی که دوری و دوستی شود، خود کلید خارجی باید ستون مشترک بگیریم، برای اینکه بدونی کدام را با کدام بگیری لازم نیست آزمون و خطا کنی از اون وابستگی بخشی دمش را بگیر که میشود a یعنی باید دم آدم بد که وابستگی بخشی ها را ببریم بعد که انداختیم دور دم میماند و جای c که مانده هست b میاد وقتی دم را میگیری پرت میکنی بیرون دیگر وابستگی میشود کامل چرا چون اونجا عضو کلید بود اینجا تمام کلید هست. دقت کن واسه قسمت دوم باید قانون ارسطو بزنیم و بستار بگیریم برای کلید همینجوری مشخص نمیشود. دقت کن عنصر تکراری داشت یکبار بیشتر نمینویسیم بعد از همه اینها دقت کن هر موقع 1 دیدی باید اونور 2 باشد یا هر موقع 7 باشد باید اونور 3 ببینی یعنی وابستگی تابعی دارد ولی از نوع کامل. چرا تکراری داریم مثل 2 چون a تکرار شده هست که 1 هست و نیازی به نگرانی نیست چون a کلید نیست پس میتواند تکرار شود بخاطر همین هست که وابستگی بخشی هست ولی وابستگی تابعی کامل تکرار به وجود نمیاورد چرا ؟ چون سمت چپ کلید کاندید هست و تکرار نمیشود چون کلید باید یکتا باشد پس سمت راست هم تکرار نمیشود ولی در وابستگی بخشی، عضوی از کلید هست و خودش کلید نیست. مشخص هست دیگه چون وابستگی تابعی کامل دارد به ازای 1 فقط میدهد 2 و تکرار رخ نمیدهد. شرط لازم بر قرار هست چون کلید خارجی تعریف شده و بی کاست و بی حشو هست برای این مثال و دقت کن سایر نگاشت ها نسبت به همین دم را استفاده کنیم رکورد های جدید اضافه میشود هنگام الحاق و شرط لازم از بین میرود. آقای هیث بخشی را حذف میکند. یعنی ما فهمیدیم کلید این جدول ab هست بعد فهمیدیم ac وابستگی بخشی دارند و وابستگی بخشی چیز بدی هست و دم را میگیریم میندازیم بیرون و شرط لازم برقرار هست ولی آیا شرط کافی هم هست؟ یعنی حافظ FD باشد یعنی ab->c باقی مانده است؟ بله چون دست به چیزی نزدی و کلید خارجی هم تعریف کردی یعنی این حافظ FD بودن خود به خود هست.

مثلا در بقیه تجزیه ها که درست نیستند حافظ FD بودند حذف شده است دقت کن برای تعریف کلید خارجی یک طرف باید کلید کاندید باشد R12 کلید کاندید هست و R11 کلید خارجی ستون a.

در بقیه حالات ممکن هست تکراری باشد. کلید خارجی باید در یک طرف کلید کاندید باشد. فقط دم بخشی را بکش بنداز بیرون همین.

**3nf**: در فرم سوم برای وابستگی انتقالی رفع مشکل میکنیم. اول باید کلید پیدا کنیم ببینیم جدول حالش خوب هست یا نیست دقت کن که نباید از روی محتوا نظر بدهیم، یعنی اول باید فرمول را بزنیم که چی بود ستون ها منهای سمت راستی ها بعد عضو کلید کاندید بدست میومد بعد باید بستار میزدیم ببینیم همش میاد یا نه بعد قانون ها را میزنیم که مثلا در مثال اول چون تهی نشده است پس قانون چهارم نیست و چون همه را میدهد قانون دوم هست و تنها کلید کاندید ما a میشود. غیر کلید، غیر کلید را بدهد میشود **وابستگی انتقالی**، دقت کن در نرمال سازی نباید به چیزی دست بزنی تا وابستگی ها خراب نشود تا حافظ FD بمانی یعنی به صورت اتوماتیک هم بی کاست و بی حشو میماند هم حافظ بودن بعد دم آدم بد را میگیریم میندازیم بیرون ولی جاش میماند که همان منظور وابستگی انتقالی هست. بعد دقت کن تجزیه کردی اول باید ببینی ستون مشترک چیست بعد حتما باید کلید خارجی تعریف شود که سر فلش میشود کلید کاندید ببین اینجوری فکر کن اونی که اینجا غیر کلید هست و سر دم آدم بد هست بعد از تجزیه خودش در همینجا میشود کلید خارجی و در جدول حاصل شده از تجزیه میشود کلید کاندید. ستون مشترک کافی نیست بلکه باید کلید خارجی هم تبدیل شود. هر جا هم گیر کردی در کلید عضو کلید کاندید بزن بعد بستار بگیر تا شک برطرف شود. و هر جور دیگر رفتار کنی رکوردی به جدول اضافه میشود دقت کن نباید رکوردی به جدول کم یا زیاد شود. دقت کن میشود وابستگی بخشی داشته باشد هم انتقالی و دقت کن اول بخشی حل کن بعد بیا انتقالی. وقتی وابستگی بخشی نداشته باشد 2nf هست.

**فصل نهم بخش سوم:**

سر فلش به دمش یا اولش وابسته هست یعنی a->c یعنی c به a وابسته هست یا a میدهد c را. دقت کن چون چپ مشکل دارد باعث تکرار میشود. اگر وابستگی کامل باشد تکرار ندارد. ولی اگر مشکل داشت باشد یعنی در جدول دوم هر موقع 2 بیاد باید 5 بیاد یعنی همون وابستگی. وقتی یک سطح از نرمال میکنی تکرار های حاصل از اون سطح حذف میشوند و تکرار های باقی میمانند مثلا 1nf نرمال میشود ولی بخشی و انتقالی تکراری هاش میماند. دقت کن اول باید وابستگی را بدهد تا بعد نرمال سازی کنیم، چون وابستگی نداشته باشیم انگار کلید نداریم. 2 صورت میشود وابستگی ها را نشان داد نوشتاری که همین هست که تا الان دیدیم و شکل نموداری. هر وقت مستطیل دیدی کلید نیست دقت کن. مستطیل نشانه وابستگی یا با هم بودن هست. یعنی s# و p# با هم qty را میدهند. یا city میاد status را میدهد. مربع نشانه تک صفت هست و مستطیل یعنی با هم بودن. به تعداد یال ها رابطه داریم. شرط کافی همیشه همراه هست با عمل تجزیه. چرا دمش را گرفتیم انداختیم اونور باید جاش بماند؟ چون اونجا قرار هست کلید خارجی تعریف شود. هم کلید خارجی تعریف میشود هم وابستگی ها حفظ میشود. دقت کن جداول تجزیه شده باید با الحاق طبیعی دوباره به هم متصل شوند شرط هم ستون مشترک هست که اگر نباشد میشود ضرب دکارتی. دوباره میگم ستون مشترکی که دمش را گرفتی کندی در جدول اولیه کلید خارجی و در جدول دومی حاصل از تجزیه کلید کاندید هست. دقت کن وابستگی بخشی چرا ایراد دارد؟ چون وابستگی افزونه ساز هست یعنی اینکه هر موقع s1 اومده بعدش A اومده که افزونگی الکی هست یا همین برای ستون S# و status که هر موقع s1 بوده وضعیت ما 2 هست به نوعی الکی تکرار شده چون چپ خوب نبوده و تکرار شده و باعث تکرار سمت راستش هم شده است. کلید خارجی میتواند تکرار داشته باشد. دقت کن اگر a->b مشکل داشته باشد ما زائده را وقتی دمش را میگیریم میکنیم انگار از روی a برمیداریم پس طبیعی هست که جای آن بماند.

دقت کن وقتی تجزیه کردیم و مشکل 2nf حل شد دوباره باید برویم کلید در بیاوریم از جداول حاصل شده، وقتی هم وابستگی کامل باشد و وابستگی دیگری نداشته باشد 3nf هم هست. در واقع چون چپ مشکل دارد باعث تکرار سمت راست هم میشود و خوب نیست.

دقت کن شرط الحاق ستون مشترک هست اگر نبود تبدیل میشود به ضرب دکارتی، دقت کن برای بی کاست و بی حشو بودن باید کلید خارجی تعریف کنیم، خوب وقتی دم را میگیریم برمیداریم اتوماتیک وار چون جاش میماند کلید خارجی تعریف میشود و به وابستگی تابعی دست نمیزنیم و حافظ FD میشویم به این میگویند تجزیه مطلوب. دقت کن وقتی C->B مشکل داشته باشد و دم را بگیریم بکنیم از روی C میکَنیم. بعد دقت کن وقتی میکنیم میبریم status میرود دیگر نمیماند که S# بدهد status و فقط city که دم هست میماند. دقت کن اجتماع وابستگی های بعد تجزیه باید جدول پایه را تشکیل بدهد. دقت کن تعداد برگ ها میشود تعداد جداول ما در درخت تجزیه.

فرم نرمال BCNF: دقت کن سمت چپ کلید کاندید باشد چون تکرار تولید نمیکند پس چیز خوبی هست. اگر سمت چپ ابر کلید هم باشد همین هست یعنی تکرار تولید نمیکند یعنی هم FFD هم FD تکرار تولید نمیکنند. برعکس بخشی میشود وابستگی معکوس یعنی غیر کلید عضو کلید را بدهد. وابستگی معکوس هم افزونگی تولید میکند. اگر یک عضو کلید یک عضو کلید دیگر را بدهد هم معکوس حساب میشود. اگر همه شروع های وابستگی ها ابر کلید باشد آنگاه آن جدول قطعا BCNF هست. چه وقتی همه شروع ها ابر کلید هست؟ یا ابر کلید باشد یا کاندید باشد. اگر سمت چپ ابر کلید باشد سمت چپ تکرار تولید میکند؟ خیر. چه زمانی همه شروع ها ابر کلید هست؟ وقتی بخشی و انتقالی نداشته باشی. دقت کن جدولی هست که همه شروع هاش ابر کلید باشد و BCNF باشد ولی 4NF نباشد؟ بله داریم. جدولی که تمام کلید هست قطعا BCNF باشد ولی 4NF نباشد. دقت کن این تعریف برای تشخیص هست نه حل آن. دقت کن این تعریف بخشی و انتقالی هم درون خود دارد و اگر همه شروع ها وابستگی ابر کلید نباشد آنوقت 100 درصد در BCNF قرار ندارد.

**فصل نهم بخش چهارم:**

دقت کن به صرف اینکه بگوییم جدولی وابستگی انتقالی ندارد نمیتوانیم بگوییم 3nf هست بلکه باید شرط لازم که 2nf بودن هم هست داشته باشد. تازه نباید چند مقداری مرکب هم داشته باشد. اگر جدولی وابستگی معکوس نداشته باشد نمیتوانی بگویی که BCNF هست دقت کن، هر جدول دو ستونی 100 درصد BCNF هست. وقتی میگیم همه شروع ها ابر کلید باشد یعنی وابستگی معکوس هم چک میکند تازه بخشی و انتقالی هم چک میکند. دقت کن اگر معکوس نباشد به این معنی نیست که BCNF هست چون باید ببینی 2NF 3NF هست یا نیست دقت کن هم FD و FFD را تیک و ضربدر میزنی چون دنبال ابر کلید ها هستی. وقتی BCNF باشد از 3NF به قبل را هست دیگر لازم نیست چک کنی. قانون سوم 2 تا کلید دارد و ستون مشترک هم دارد. دقت کن ما از روی وابستگی کلید بدست میاوردیم نه که با کلید بریم دنبال وابستگی اینا. آیا هر جدولی که در قانون سوم هست BCNF هست؟ خیر. کیا تو قانون سوم هستند؟ اونایی که بیشتر از یک کلید دارند و این کلید ها با هم وجه اشتراک دارند. اگر چیزی برود تو قانون سوم پتانسیل BCNF نبودن را فعال میکند چون کلی عضو کلید دارد و بحث معکوس بودن به وجود میاد. البته دقت کن اگر قانون سوم نباشد باز نمیتوانی بگویی BCNF هست چون باید بخشی و انتقالی هم چک کنی. سر فلش سمت راستی ها هستند. قانون چهارم شد یکم آزمون و خطا کن تازه تو کنکور گزینه ها را میدهد ببین کدوم ها میشود و کدام کلید میشود چون کلید اشتراک ندارند و ببین کدوم ها همه را میدهند. هر جدولی که تو قانون سوم باشد و معکوس هم داشته باشد BCNF نیست. و قانون سوم باعث ایجاد وابستگی معکوس میشود. جدولی که 3NF باشد و یک کلید داشته باشد پس BCNF هست چون عضو به عضو وجود نمیاد اصلا مثل قانون سوم. جدولی که 3NF باشد و در قانون دوم باشد حتما BCNF هست. قانون دوم وابستگی معکوس را نابود میکند. اگر کلید با قانون 4 و 2 کشف شه و 3NF باشه حتما BCNF هم هست.

هر جدولی با 2 ستون حتما BCNF هست. جدولی که تو قانون سوم باشد ولی وابستگی معکوس نداشته باشد حتما BCNF هست. دقت کن در قانون چهارم یک کلید بدست آمد بعد بازگشتی کن یعنی چی یعنی مثلا اگر کلید AB بود بگو کی دارد A را میدهد؟ C کی دارد B را میدهد؟ خوب D پس ترکیب این 2 هم همه را میدهد پس میشود کلید. دقت کن تو بستار از چیز هایی که تا الان بدست آوردیم برای کشف بقیه ستون ها هم کمک میگیریم. مثلا در A,D ما از AB برای بدست آوردن C استفاده کرده ایم. دقت کن در قانون چهارم برای همه کلید ها میگوییم اشتراک دارند یا خیر.

**فصل ششم بخش اول:**

DDL ظرف میسازیم با DML محتوا میسازیم، با DCL حق دسترسی تعیین میکنیم که مثل دستورات امنیتی یعنی کی میتواند فلان چیز را ببیند. سلکت جبر سطر انتخاب میکرد اینجا ستون انتخاب میکند. در جبر رابطه ای سطر های تکراری خود به خود حذف میشدند ولی اینجا حذف نمیشود، یعنی خیلی از جبر رابطه ای پیروی نمیکند. دقت کن تکراری ها حذف نمیشوند. اگر distinct بزاریم تکراری ها حذف میشود. Where اختیاری است. All که متضاد distinct هست به صورت پیش فرض هست ننویسیم در نظر میگیرد خودش. S,P یعنی ضرب دکارتی در SQL، الحاق طبیعی اینها را باید شبیه سازی بکنی اگر خط where نباشد همه را میدهد حتی پیوند ناپذیر ها ولی where بزاری فقط پیوند پذیر میدهد نمیشه بگیم هیچ موقع پیوند ناپذیر بدرد نمیخورد چون مثلا میگوید شماره تولید کنندگانی که هیچ قطعه ای تولید نکرده اند. پیوند پذیر هم که میدونی بر اساس ستون مشترک و مقداری که در آن دارد ضرب میشود در ستون مشترک اون یکی جدول در مقدار مشترکی که هر دو ستون دارند. در ضرب دکارتی ستون مشترک دوبار میاد در الحاق هست که یکبار میاد. دقت کن sql اول به این معنی نیست که s1 اومده و p6 را تولید کرده است نه فقط میگوید همشهری هست چون جدول SP که نیست اصلا. کلمه تولید کننده به این معنی نیست که حتما قطعه تولید کرده است. As یعنی نام مستعار بهش میدهی البته دقت کن نام اصلی عوض نمیشود و هنوز معتبر هست. دقت کن در sql ضرب دکارتی یک جدول در خودش مورد قبول نیست. دقت کن در ضرب دکارتی تمام ستون ها حتی مشترک ها میادش کنار هم انگار جمع کردی تعداد ستون یا همان پراجکت ها را. پس اگر یک جدول را در خودش ضرب دکارتی کنی ستون مشترک را بگیری شماره تولید کنندگانی را میدهد که همشهری هستند. کلمه as را میتوانی ننویسیم. هم برای جداول هم برای ستون میتوانی از as استفاده کنیم. Like عملگر منطقی هست بدرد شباهت میخورد فرقش با = این هست که دومی میگوید عینا این میگوید قسمتی مشابه یعنی دقیقا یکسان نباشد هم مهم نیست. جلوی where میگذاریم بدرد انتخاب سطر میخورد دو نماد درصد % و – دش یا خط تیره دارد. % رشته و – کاراکتر را مقایسه میکند. C% یعنی هر چی باشد منتها اولش c باشد بعد باید شبیه باشند به این رشته. C—رشته هایی هست که 3 کاراکتری هستند که با c شروع میشود. دقت کن چیزی جز جدول هست و کاراکتر محسوب میشود مثل a مثل b و دیگر جلوی like نیست پس فرق دارد حالا اگر بخواهی خود % جستجو کنی چی؟ یک بک اسلش \ میگذاری تمام و خاصیت آنرا ازش میگیری و دیگر هر چی نیست و کاراکتر میشود. در همان مثال دومیش میگوید هر چی که با t شروع شود و بعدش هر چی یعنی اون درصدی هم میادش. به حروف کوچک و بزرگ حساس هست یعنی B یعنی فقط بی بزرگ هر چی هم که تا الان گفتیم not like بگذاریم انتخاب نمیکند فقط. اگر بخواهیم بگردیم null هست از is null کمک میگیریم، برای null نبودن میگوییم is not null. Is فقط برای نال هست. جلوی where 3 چیز هست مساوی like و is null. Union میشود اجتماع خودمان. دقت کن اجتماع میگیریم دلیلی ندارد جدول ها یکی باشد فقط هم تیتر باید باشند. اگر خروجی متفاوت بود نمیتوانستیم اجتماع بگیریم ولی وقتی خروجی یکی هست هر دو select یکسان دارند میتوانیم اجتماع بگیریم. دقت کن اجتماع گرفتی تکراری ها حذف میشوند.

و به distinct هم نیاز نیست چون خود اجتماع میفهمد نباید تکراری باشد. اگر میخواستیم تکراری ها بیاد چی ؟ از union all استفاده میکنیم. عملگر intersect یا همان اشتراک خودمان: دقت کن اینجا هم جدول میتواند متفاوت ولی خروجی یا select باید هم تیتر و یکسان باشند. اگر در کنکور از دیتا حرفی نزد همنام بودن را هم نوع بودن بگیر. دقت کن حاصل اجتماع اشتراک تفاضل تکراری ندارد ولی موقع اول که محاسبه هست تکراری میتواند بیاید. اگر میخواهی همان اول هم تکراری نباشد در محاسبات اولیه تا قبل اشتراک distinct بگذار حاصل فرقی ندارد. اگر intersect all میگرفتیم چی ؟ اون تکراری های مشترک هم میاد. دقت کن مجموعه ما نظم ندارد و مقایسه محتوایی صورت میگیرد و اصلا میتواند سطر ها بهم ریخته باشد. در واقع میشود min تعداد تکرار در سمت چپ و تعداد تکرار در سمت راست. دقت کن حاصل پرس و جو میتواند تکراری باشد ولی داخل عملگر دیگر نباید تکراری بیاد. دقت کن محاسبه اشتراک حالت مجموعه هست و محتوایی مقایسه میکنیم نه سطری. میتونی بگی زوج های دوتایی باید داشته باشند و تکی ها بدرد نمیخورد در اشتراک intersect all. در union تکراری ها همه میان و مجموع آنها میشود.

**فصل ششم بخش دوم:**

Except همان تفریق هست. بدون all یک a باشد همه a های جدول اول را میزند ولی وقتی all باشد فقط به تعداد a های جدول دوم از جدول اول کم میکنیم و سایر a ها در خروجی نمایش داده میشوند. دقت کن فقط عملگرهای مجموعه ای اجتماع و اشتراک و تفریق تکراری ها را میفهمند و حذف میکنند. تازه union all هم نمیفهمد تکراری را. Distinct سطر تکراری حذف میکند کل سطر با هم نه ستون، برای select هست اصلا و برای هیچ ستونی نیست. دقت کن اگر distinct نباشد و اون سه تا عملگر نباشند اصلا سطر تکراری حذف نمیشود! ، عملگر in و between که باید جلوی where قرار بگیرند، در عملگر between دقت شود که فضای باز هست و بالا و پایین حد هم شامل میشوند (مثلا در مثال خود 16 و 19 هم جز آنها هستند)، دقت کن ببین دیتا تایپ هم چی هست اگر عدد صحیح تعریف کردی 16.1 نیست ولی اگر اعشاری تعریف کردی 16.1 هم هست. عملگر not between هم داریم یعنی هر چی باشد غیر از بازه 16 تا 19 مثلا اینطوری بگیم قبل 15 و بعد 20 که خود 15 و 20 جز آنها هستند. in: تعلق دارد انگار در قبلی جز بازه بود ولی این فضای باز نیست یعنی فقط 12 و 17 یعنی فقط این 2 نه بین این 2 فقط و فقط این 2 عدد یعنی مقدار اون صفت متعلق باشد به این دو عدد. Not in هم داریم که میشود هر چی غیر از این دو. دیتا تایپ مهم هست که در چی جستجو میکنیم. Or را میتوانی جای in استفاده کنی. دقت کن توسط جبر رابطه ای همه پرس و جو ها را میتوانی بنویسی به جز بازگشتی ها و وقتی گفتند همه قطعات یا کلا همه یعنی تقسیم. دقت کن وقتی دنبال تولید کنندگان هستی باید بروی داخل جدول sp. وقتی گفتند حداکثر 1 یعنی صفر هم هست. حداکثر 2 یعنی 0 و 1 و 2. حداقل 1 یعنی کف 1 هست و بیشتر میشود. جبر چک نویس و SQL پاک نویس هست. معیار این هست پرس و جو ها باید تمیز باشند و زمان کمتری بگیرند و حافظه کمتری هم بگیرند. پرس و جو های نرمال که نوع 1 هستند چه ویژگی دارند؟ در نرمال ها خارجی (بیرونی) به داخلی (داخل پرانتز بعد where) وابسته هست. خارجی از داخلی خوراک میگیرد کلمات هم مثل in و ... هست، از نظر مرتبه اجرایی همیشه in از مرتبه o n + m هست پس وقتی خارج به داخل وابسته هست پس اول باید داخلی اجرا شود مشخص هست ولی در گروه دوم چون داخل به خارج وابسته هست اول باید خارجی اجرا شود. در نرمال ها یا گروه 1 کلا یکبار پرس و جو داخلی اجرا میشود 1 بار برای همیشه. Where سطر به سطر حرکت میکند. دقت کن چون in حالت تعلق دارد پس عین عملگر مجموعه ای باید سطر ها همنوع و هم تیتر باشند. چرا مرتبه اجرایی شد o n + m چون دو تا جدول داشتیم که هر 2 جدول به تعداد تمام سطر هاش اجرا شد. در عملگر in پرس و جو داخلی فقط یکبار اجرا میشود دقت کن. دقت کن اگر گروه دوم بودند به ازای یکبار خارجی یکبار داخلی اجرا میشد و میشد o n \* m. در اینجا خارجی از داخلی خوراک میگیرد.

**فصل ششم بخش سوم:**

دقت کن در ضرب دکارتی اول کل دو جدول با هم ضرب میشود بعد اتصال برقرار میشود، where جواب ترو و فالس میدهد یعنی میگوید در جدول s اولین s# در حاصل اون کوئری داخلی هست یا نیست اگر بود ترو و سطر را میدهد اگر نبود یعنی فالس و سطر را نمیدهد. دقت کن هر دو به تعداد سطر های خودشان اجرا میشود ولی به ازای اجرای یک خارجی یکبار داخلی اجرا نمیشود چون حلقه نیست نرمال هست. برای کوئری نوشتن به محیط زیست فکر کن مثلا اول فکر کن s1 مال کدوم شهر هست بعد بقیه رو از روی آن بسنج. از آخر به اول چون خوراک از آخر به اول هست. مساوی همین in هست چون یک مقدار دارد select این قضیه همیشگی هست و مفهوم ساختاری هست دقت کن منظور جلوی select نیست منظور این هست که select همیشه یک چیز را میاورد مثل شهر ولی ممکن هست چند قطعه باشد و اینجا نمیتوانی از مساوی استفاده کنی. راه حل سوم: exists: جز گروه دوم هستند، از لحاظ عملکرد بهتر in بهتر از exists هست. و در اینجا داخلی به خارجی وابسته است و داخلی از خارجی خوراک میگیرد پس اول خارجی باید اجرا شود. نگاه به خارج دارد و با خارجی حرکت میکند. و مشخص هست که s از بیرونی دارد تغذیه میشود و دقت کن از خارج شروع میشود. یک نکته مهم این است که به ازای هر بار که خارجی یک سطر را میاورد یک بار هم داخلی اجرا میشود بر خلاف قبلی که اول داخلی کامل اجرا میشد بعد خارجی سطر به سطر اجرا میشد و خود را چک میکرد این مال گروه اول هست الان به ازای هر سطر خارجی یکبار داخلی اجرا میشود. دقت کن s# هر بار توسط خارجی مقدار دهی میشود بعد اجرا میشود. پس 3 روش شد 1. ضرب دکارتی 2. In 3. Exist که جز گروه دوم بود. Not exist وقتی ترو میشود که جلوی آن تهی باشد وقتی جلوی آن پر باشد دیگر ترو نمیشود. دقت کن سطر وقتی انتخاب نمیشود که where فالس شده باشد خود where کی فالس میشود؟ وقتی not exist فالس شود. unique کی استفاده میکنی؟ هر موقع فقط اومد، مثلا فقط یک قطعه تولید کرده باشد در نتیجه s1 که 2 تا تولید کرده خط میخورد. فقط 1 و not exist یعنی 0. عملگر some یا any: بعضی ترجمه کن، یعنی مثلا وزن مساوی با بعضی از ایناست مرتبه اجرای n + m خارجی به داخلی وابسته هست و نرمال هست یا گروه اول.

**فصل ششم بخش چهارم:**

**عملگر all:** نرمال هست و مرتبه آن مشخص هست. میگوید از همش یعنی مثلا از همه اون داخلی باید کوچکتر یا بزرگتر باشد. دقت کن where دومی روی temp صورت میگیرد که حاصل جدول from از sp هست. جدول میانی همان temp هست کلا این جدول ها جایی ذخیره نمیشوند. جدول مجازی همان view ها هستند. **group by**: ترتیب هم دقت کن جدول، سطر و ستون. این عملگر قبل از سلکت هست. به ترتیب ها هم دقت کن. از نظر سینتکس سلکت باید زیر مجموعه group by باشد یعنی اون ستون هایی که قرار هست انتخاب کنیم طبیعتا باید در group by باشد اول. سلکت در دل گروپ بای هست یعنی بزرگتر از آن نمیتواند باشد. اگر گروپ یا تابع آماری یا order by داد را بزار کنار، و برو سراغ where اون سطر های که شرط را ارضا میکنند بیار بعد گروه بندی میکنیم مثل زن ها و مردان که گروه بندی میکنیم، سطر های حذف شده دیگر اعمال نمیشود. بعد میای گروه بندی میکنی اون ستون ها را که خواستی مینویسی بعد بیرون پرانتز مینویسی با کیا آمده است. یک گروه s1,12 بعد یک گروه دیگر s1,13 به تعداد هر کدام از مختلف ها گروه بندی میکنیم بعد بین اینها سلکت میکنیم و هنگام سلکت فقط سر گروه ها را مینویسیم و دیگر اینکه با چه ستون هایی آمده اند نداریم. Distinct که میدانی تکراری حذف میکرد. گروپ بای به نوعی انگار distinct هست چون تکراری ها را فقط سر گروه مینویسد. **توابع آماری**: مثلا ماکس یا مین یا تعداد و ...، اولین count هست. **count**: تعداد مقادیر یک ستون را میشمارد و تعداد شمار هست دقت کن اینجا ستون برنمیگراند چون سلکت خالی نیست بلکه فقط تعداد برمیگرداند تابع count نال ها را نمیبیند و نمیشمارد. جدول را وقتی اول میسازی اول خالی هست وقتی create میکنی. بعد از اولین نمونه شروع میکند اعمال کردن رشته ها را هم میشمارد فقط عدد برمیگرداند و فقط نال را نمیشمارد تکراری هم میشمارد. دقت هم کن فقط یک ستون. **تابع count \***: تعداد سطر های یک جدول را میشمارد یا سطر شمار هست. از سطر null میپرد کل سطر باید null باشد. **تابع AVG**: میانگین مقادیر یک ستون را محاسبه میکند null باشد از اون میپرد و در میانگین نه جز تعداد تقسیم میشمارد نه جز مقادیر حاصل جمع دقت کن این دیگر تعداد برنمیگرداند و اگر نتواند میانگین برگرداند null میدهد چون میانگین نگرفته که 0 برگرداند فقط count عدد میدهد. روی رشته صورت نمیگیرد و ارور میدهد. میانگین فقط روی عددی ها کار میکند. **تابع max**: بزرگترین مقدار یک ستون دقت کن اینجا هم null را null میدهد تنها جایی که null میبیند 0 میدهد count هست دقت کن با کد اسکی کار میکند و روی غیر عددی ها هم کار میکند و b از a بزرگتر است و برمیگرداند. دقت کن 33 جز رشته هست و در رشته کاراکتر به کاراکتر میرود جلو مقایسه میکند هر جا تفکیک شد بزرگ را بنویس. Min برعکس همین max هست.

**فصل ششم جلسه پنجم:**

از سر توجه: نوشتن توابع آماری با where خطا نحوی دارد، اصلا کامپایلر اجرا نمیکند و سینتکس ارور دارد پس where مستقیم جلوی توابع آماری نمیاد و اگر میخواهد نوشته شود باید با یک select نوشته شود. توابع آماری جلوی سلکت نوشته میشود و توابع آماری اصلا صاحاب ندارد که مال کدام جدول هست. دقت شود که اول داخل سلکت 30 را برمیگرداند بعد سطر به سطر حرکت میکند و هر کی مساوی 30 بود را برمیگرداند. گاهی اوقات ممکن است بخواهد هر چی بدهد جز ماکس یا هر چی از ماکس کوچکتر هست در خروجی باشد. تو کوئری ها فلش بگیر سطر به سطر برو پایین. با some بیاد یعنی اونایی که از بعضی هاش کوچکتر هستند در ماکس مثلا 30 از بعضی ها کوچک نیست ولی 20 یا 10 از بعضی های این کوچک هستند همونطوری فلش بزار سطر به سطر برو پایین. همین در min میشود که از بعضی ها کوچکتر باشد مثلا 10 از بعضی ها کوچکتر نیست. دقت کن مین با مساوی یا ماکس با مساوی، مین و ماکس را میدهد. all میاد یعنی همش یعنی میگوید از این ها که سلکت کردیم و سطر به سطر حرکت کن بگو 20 از همش؟ 10 از همش؟ 30 از همش؟ بزرگتر هست یا نیست. همین برای min. دقت شود distinct به ستون میچسبد همیشه. همیشه جلوی count باید ستون باشد. جلوی distinct هم باید ستون باشد و برای همین count چون عدد میدهد نمیتواند جلوی distinct باشد. وقتی distinct میزاری تکراری ها حذف میشوند و count بدون تکراری ها را میشمارد. As بگذاری نام میدهد به اون ستون و حتما در خروجی count هم لحاظ میکند و دیگر no column name نمینویسد. برای count\* از distinct نمیتوانیم استفاده کنیم. تابع آماری دادن بزار کنار اول سلکت کن بعد بین اونا بردار بیار. ستون سلکت باید زیر مجموعه ستون گروپ بای باشد و تابع آماری min یا max یا هر چی ستون نیست. دقت کن گروه بندی چیزی را حذف نمیکند. تنها تابعی که همیشه عدد میدهد count هست. اگر گروپ بای با تابع آماری بود اول گروپ بای بزن دقت کن قبلش شرط زدی و اونا که نیستن افتادن با where بعد بیا یک جدول دو ستون بکش یکی که s# هست هیچی قبلا هم بود یکیش اما no column name هست که میشود همان min. بعد دقت کن توابع آماری روی گروه ها اعلام میشود یعنی میگوید گروه s1 کمترین را سلکت کن و ... ، به این درد میخورد که مثلا یک شرکت داریم 3 تا دپارتمان دارد میخواهیم کمترین حقوق بین این دپارتمان ها را پیدا کنیم. دقت شود که وقتی گروه بندی میکنیم توابع آماری روی گروه ها اعلام میشود یعنی count تعداد اعضای گروه را میشمارد مربوط به s1 و ...، sum و avg روی ستون های عددی کار میکنند و sum p1 معنی نمیدهد. ولی مثلا sum QTY معنی میدهد. عملگر having: انتخاب کننده گروه هست برخلاف where که انتخاب کننده سطر هست. مثلا گروه خانم ها و آقایان میاد و ما فقط یک گروه را میخواهیم ببریم اردو مثلا پس باید از دستور having استفاده کنیم پس نمیشود بدون گروه بندی از آن استفاده کرد. Having فقط یک گروه را انتخاب نمیکند دقت کن هر چی که شرط را ارضا کند انتخاب میکند مثلا where هست همه چیز آن فقط این سطر هست اون گروه. دقت کن شرط تابع آماری جلوی having میاد. به شماره ترتیب ها دقت کن چون قبل having باید گروه بندی باشد. دقت کن هر پرس و جویی دیدی هم گروپ بای داشت هم سلکت اول قانون زیر مجموعه بودن سلکت را بررسی کن. عملگر order by: اول دقت کن باید زیر مجموعه سلکت باشد، دقت کن اگر مقادیری که میخواستی ترتیب بندی کنی مساوی بود بر اساس ترتیب جدول و محتوا بچین. دقت هم کن اول برو تو جدول where رو بزن با تیک و ضربدر بعد هر بلایی خواستی سرش بیار این شرط همه پرس و جو ها هست. جلوی سلکت میتوانیم محاسبات بگذاریم و ستون آن شبیه به ستون های sum, count, avg هست. order by روی سلکت کار میکند نه جدول پایه، اگر هیچی نگوید مرتب سازی صعودی هست و دقت هم کن که در اون مثال دارد روی 2 ستون اعمال میشود. مرتب سازی همیشه از چپ به راست هست یعنی اول وزن مثلا بعد p# دقت کن همزمان با هم مرتب سازی نمیکند هر جا که گیر میکند از p# کمک میگیرد. ستون ساختاری نام آن و ستون محاسباتی بدون اسم میاد دقت هم کن صعودی اول p1 بعد p5 اگر هم نگفته بود بر اساس محتوا. 2 همان weigh هست. پیشفرض ASC هست. دقت کن اگر یکبار ASC یا DESC را به کار ببری برای هر 2 تا هست ولی اگر بخوای تیکه به تیکه بکنی باید برای هر کدوم بگی.

فصل ششم بخش ششم:

در except دو طرف باید هم تیتر باشند. وقتی جلوی not exist تهی باشد ترو میشود و تهی نشود فالس قرار میگیرد پس where هم ترو میشود. همیشه تقسیم یا با not exist و not exist هست یا با not exist و except هست. 4 تا حالت داریم واسه 2 تا not exist یعنی یه بار جفت همین یه بار یکی exist اون یکی not exist یکی جفت exist. نباشه که تولید نشده باشه اینطوری بخوان که معنی میدهد همه تولید شده باشن تولید واسه دومی هست. دقت کن where ما را روی from حرکت میدهد. دقت کن شرط اتصال که مال پرس و جو داخلی هست از بیرون خوراک میگیرد مال پرس و جو های شماره 2 و خودش به تنهایی حرکت نمیکند و منتظر where بالایی میماند. و برای سطر s1 یکبار برای p1 میروی حساب میکنی یکبار برای p2. بعد چون where دوم تهی شد کلا not exist اول ترو و where کنار آن هم ترو میشود پس s1 میاد بیرون حالا میرویم برای s2. دقت کن این پرس و جو واسه تولید کنندگانی هست که همه قطعات را تولید کرده باشند. مثل سور ها رفتار کن.

**ادامه دستورات DML**: دستور insert: این با delete و update دستورات DML هستند. در inset به دیتا تایپ ها دقت کن که رعایت شود اونایی که مقدار دهی نکرد null میشود. دقت کن میتوانیم نظم ستون ها را بهم بزنیم و به ترتیب ننویسیم و البته دقت کن باید به همان ترتیب هم مقدار دهی کنی. اگر هیچی جلوی جدول مورد نظر ننویسیم باید نظم پیش فرض جدول را رعایت کنیم و دلخواه نیست. وقتی چیزی نمینویسی جلوی جدول یعنی کل ستون ها را میخواهی و باید اگر 5 تا ستون هست 5 تا مقدار دهی بکنی اگر با رعایت دیتا تایپ نمیخواهی یک ستون را مقدار بدهی باید کاما بگذاری، تا معنی null را بدهد. تعداد ویرگول هم یکی کمتر از درجه جدول ما هست. میتوانی مستقیم هم null بگذاری جای کاما. دقت کن کی نمیگذارد type miss match اتفاق بیفتد؟ جامعیت داخلی تعیین میکند DBMS جامعیت داخلی هم که میدانی 4 تا بود موجودیت، ارجاعی، درون رابطه ای، دامنه ای که چهارمی میگوید دیتا تایپ ها باید رعایت شود. اولی میگوید کلید نباید نال باشد. ارجاع میگوید ارجاع نال نباید باشد و درون رابطه ای میگوید جدول باید کلید داشته باشد آدم باشد و مرکب نباشد. جامعیت موجودیت میگوید کلید نباید نال باشد. دقت کن دو تا کلید تکراری نباید داشته باشیم که بدیهی هست اگر نباشد جامعیت درون رابطه ای رد شده است. قانون جامعیت ارجاعی: مقادیر کلید خارجی باید زیر مجموعه مقادیر کلید کاندید باشد و اگر نباشد قانون جامعیت ارجاعی نقض شده است کلا کلید خارجی باید از دو منظر ساختاری و محتوایی بحث بشود که محتوایی را در بالا اشاره کردیم. دقت کن اگر 2 تا کلید خارجی داریم هر 2 تا باید صاحب داشته باشند و جامعیت ارجاعی را رعایت کنند. **دستور update**: همچنان در DML ها هستیم. بروز رسانی همزمان میتواند در چند جدول باشد، با set میاد. هر وقت قوانین جامعیت ها نقض شود آنومالی یا بی نظمی شکل میگیرد. در update دقت کن باعث شکل گیری ارجاع نال نشود تا جامعیت ارجاعی نقض نشود. البته خود DBMS به صورت خودکار میاد تبدیل میکند S1 به S11 را تا ارجاع نال به وجود نیاد. در مثال بعدی برای ضرب دقت کن انتساب هنوز انجام نشده است پس میشود 10\*0.5 بعد دوباره ضربدر 10 زیرا QTY که هنوز انتسابی نداشته است و هنوز 10 مانده است. update میتواند با case بیاد که مثل سوییچ کیس عمل میکند یعنی چند تا مورد داریم به ازای مقادیر مختلف مقدار دهی میکند.

**دستور delete**: سطر ها را دیلیت میکند. دقت کن دستورات DML اصلا حق ندارد راجب DDL و ظرف ها دستکاری بکند و جدول را وقتی تمام سطر ها را پاک میکنیم درست است تهی میشود ولی در کاتالوگ سیستم باقی میماند. فقط از نظر تعداد رکورد 0 میشود. پاک کردن تمام رکورد ها هم وقتی هست که WHERE نگذاری. دقت کن تابع آماری میتواند جلوی where بیاد به شرطی که داخل یک سلکت دیگر باشد دقت کن وقتی از یک جدولی میخواهی تابع آماری بگیری و بعد در همان جدول دیلیت کنی باید این را در نظر داشته باشی که where ما وقتی میخواهد روی سطر ها حرکت کند چون با یک جدول جدید روبرو میشود ( چون یک سطر حذف شده است) پس باید دوباره میانگین بگیری و دوباره حذف و اضافه کنی و ممکن است بیشتر از یک سطر حذف شود. دقت کن با وجود اینکه نرمال کوئری هست ولی چون جدول عوض شده است دوباره اجرا میشود سلکت داخلی. این خطای معنایی دارد خطای سینتکسی ندارد و کامپایلر کامپایل میکند. برای حل این مشکل باید داخل یک temp بریزیم که این مشکل پیش نیاد. سلکت باید از نظر درجه با گروپ بای یکی باشد ولی طراحان میگویند سلکت باید زیر مجموعه گروپ بای باشد. دیتا تایپ ها را بخوان. دقت کن date مرکب هست ولی SQL دارد.

**فصل هفتم:**

دستورات DDL: domain: یا دامنه، دامنه یا رسمی هست یا غیر رسمی. رسمی ها را خود کامپایلر در اختیار ما میگذارد و غیر رسمی ها را خودمان دستی باید بسازیم.

با دستور alter میتوانی دامنه ها را تغییر بدهی و دقت کن مقادیر قبلی نامعتبر میشود. وقتی یک دامنه را ساختی میتوانی پاک هم بکنی که با drop انجام میشود با create هم که دامنه را ساختی. دقت هم شود که دیلیت در DML با delete و در DDL با drop انجام میشود. با drop کلا پاک میکند و از بین میبرد. اول باید با new database یک دیتابیس ایجاد کنی بعد مجموعه ای از جدول ها را بسازی چون یک دیتابیس از مجموعه ای از جداول تشکیل شده است. create table جدول میسازی باید حتما نام بگذاری برای جدول یا حتما باید attribute تعریف کنی مثلا not null اجباری نیست بودن یا نبودن آن. Primary key حتما اجباری هست چون جدول حتما باید کلید داشته باشد ولی سایر میتواند اختیاری باشد. یکتا بودن کلید توسط DBMS کنترل میشود که با ایجاد یک شاخص سیستمی باعث میشود که یکتایی کلید حفظ شود. شاخص یا سیستمی هستی یا کاربردی. Not null هم داخل primary key هست که این برای جامعیت موجودیت هست و یکتایی برای جامعیت درون رابطه ای پس جلوی کلید ضرورتی ندارد بگذاریم. دقت کن وقتی not null میگذاریم یعنی توسط دستور insert هم نمیتوانی null وارد اون ستون بکنی. جز not null برای ستون city که جامعیت خارجی هست و طراح بانک میگوید نباید نال باشد سایرین جامعیت داخلی هست و DBMS مشخص میکند. هر چی که ما بگیم و دیتابیس باید رعایت بکند میشود جامعیت خارجی. با drop هم جدول را پاک میکنیم. حذف فیزیکی به معنای حذف کردن از روی دیسک هست. ستونی که محدودیت not null نگذاریم میتواند مقدار null داشته باشد. دستور option: 3 فرم دارد هر کدام نوع حساسیت یا درجه حساسیت را دارند اولی no action هست که پیش فرض هست. نوع دوم cascade هست و نوع سوم set null هست. دقت کن این فرم ها باید در جداول فرعی اعمال شود چون جدول فرعی به اصلی نگاه میکند بپرس کی به کی نگاه میکند. پس sp باید حساس باشد چون sp به s نگاه میکند و به تغییرات آن حساس هست. کاتالوگ: شناسنامه هست در کتاب مرجع کاتالوگ با دیتا دیکشنری را یکی در نظر گرفته شده است. و منطقی هم هست. کاتالوگ مشخصات سیستمی جداول ما هست که بهش فرا داده یا متا دیتا گفته میشود. مشخصات کل جداول و سیستمی داخل کاتالوگ هست الی محتوا، یعنی مثلا اینکه جدول ما چند سطر دارد داخل کاتالوگ هست. از ایجاد تا حذف یک جدول کل اطلاعات آن داخل کاتالوگ هست پس با دستور delete جدول پاک نمیشود و از داخل کاتالوگ پاک نمیشود. توسط تمام دستورات DDL و برخی دستورات DML کاتالوگ دچار تغییر میشود. مثلا ساختن یک جدول یا حذف کنی. تعداد رکورد ها داخل کاتالوگ هست پس با delete و insert کاتالوگ تغییر میکند ولی سایر دستورات DML کاتالوگ تغییر نمیکند. برگردیم سر موضوع قبلی. سطح حساسیت no action: یعنی بدون حرکت مثلا s3 را s33 بکنیم تاثیری روی sp ندارد و کاری نباید بکند و اگر s1 بشود s11 دیگر در sp تغییری نمیکند چون بی تفاوت هست اون cascade هست که در جدول sp s1 را s11 میکرد ولی اینجا چون sp حرکتی نمیکند s از حرکت خودش پشیمان میشود و s11 را s1 میکند چون میداند sp کاری نمیکند و مشکل ایجاد میشود. اگر s1 را پاک کنی هم چون sp کاری انجام نمیدهد و کاری صورت نمیگیرد. Cascade: آبشاری هست s3 بشود s33 که تاثیری ندارد ولی اگر s1 بشود s11 در جدول sp هم s1 را به s11 تبدیل میکند. اگر s3 را دیلیت کنی هم اتفاقی نمیفتد ولی اگر s1 را دیلیت کنی، در sp هم دیلیت میشود. Set null: در مثال بعدی هست اگر 4 را 44 بکنی یا دیلیت کنی تاثیری ندارد و موردی ندارد در r2 و اگر کلا پاک کنی هم مشکلی ندارد و پاک میشود. ولی اگر 1 را به 11 تبدیل کنی جفت 1 ها نال میشود و دیگر به مقدار جدید اپدیت نمیشود. و اگر دیلیت هم کنی نال میکند. ولی دقت کن این برای ستون هایی بودند که کلید نبودند در ستون هایی که کلید هستند نمیتوانی null بگذاری پس همه چیز به حالت سابق برمیگردد و s11 به s1 تبدیل میشود.

**فصل هفتم بخش دوم:**

دستور alter میتوانیم جداول را تغییر بدهیم در DDL ها. دامنه جدید باید زیر مجموعه دامنه قدیم باشد و گرنه دچار از دادن داده میشویم.

مگر اینکه فارغ از داده باشد دقت کن مثلا همینطوری نمیتوانیم از استرینگ به اینتجر یا از اعشاری به اینتجر چونکه ممکن هست موجب از دست دادگی داده بشویم. مقدار not null هم نمیتوانیم بگذاریم اینها فقط برای شروع هست. جامعیت درون رابطه ای یعنی جدول بدون کلید بشود مثلا. دقت کن هر ستونی را نمیتوانی تغییر بدهی مثلا هر ستونی را نمیتوانی حذف کنی باید طوری رفتار کنی که موجب رد شدن قوانین جامعیتی نشود. اگر یک جدولی داشتیم که در آن یکی از ستون های آن که کلید اصلی هست در جای دیگر کلید خارجی باشد امکان حذف آن جدول را نداریم چرا که باعث جامعیت ارجاعی نقض میشود. دقت کن واسه alter دامنه جدید باید ظرف بزرگتری باشد نه ظرف کوچکتری مثلا از int به small int نمیشه تغییر داد چون باعث از دست رفتن اطلاعات میشود. دامنه یکی نباشد جامعیت دامنه ای است.

**Index**: یا شاخص، به مجموعه ای از جداول و view ها و پرس و جو ها و index ها میگوییم یک دیتابیس و index باعث افزایش سرعت جستجو میشود و باعث میشود در زمان کوتاه تری اون پرس و جو را انجام بدهیم و سریعتر میشود. وقتی میگی primary key یعنی یک شاخص سیستمی میاد و not null میشود و همچنین توسط سیستم گارانتی و حفظ میشود یکتایی آن، DBMS ایجاد میکند شاخص سیستمی را DBMS و شاخص کاربری را ما انتخاب میکنیم. جامعیت درون رابطه ای میگوید جدول کلید داشته باشد و مقدار مرکب نداشته باشد پیاده سازی اون گارانتی DBMS توسط همین شاخص سیستمی هست. شاخص هم میتواند یکتا و غیر یکتا باشد. دقت هم کن تاخیر کل میشود کاردینالیتی ضربدر تاخیر یک سطر. این یک شاخص کاربردی هست اگر روی city باشد. دقت کن تاثیر آن وقتی هست که روی ستونی که شاخص گذاری شده باید پرس و جو انجام شود اگر روی آن انجام نشود تاثیری ندارد و عین سابق هست. دقت کن ما برای جدول کلید اصلی تعریف میکنیم نه کلید کاندید. شاخص سیستمی فقط unique میتواند باشد ولی شاخص کاربردی میتواند یکتا باشد یا نباشد همچنین تعداد سیستمی ها 1 و تعداد کاربردی ها n تا هست. دو مدل افزونگی داریم یکی طبیعی که همان محتوایی و یکی تکنیکی هست که کلید خارجی و شاخص یعنی درست هست افزونگی دارند ولی مفید هستند. شاخص خودش را باید چون مدام باید با جدول پایه تطابق بدهد سربار ایجاد میکند و اگر نرخ تغییرات زیاد باشد چون مصرف حافظه و سربار زیادی دارد پس طراحان سعی میکنند بیشتر از 1 شاخص استفاده نکنند. دقت کن اگر یک جدول در یک ستون مقادیر تکراری داشت دیگر نمیتوانی درخواست شاخص یکتا بدهی.

**View**: جدول اصلی و میانی حافظه مصرف میکند ولی جدول مجازی که مال همین ویو هست حافظه مصرف نمیکند و از حافظه اصلی استفاده میکند. جدول میانی از کوئری ها بدست میان و موقتی هستند و امکان دستورات dml روی آنها نیست و کاربر و طراح دسترسی ندارد حالات کوئری های تو در تو هست. اطلاعات جدول مجازی درون کاتالوگ سیستم قرار میگیرد. منبع تغذیه جدول ویو جدول پایه هست. نه روی سرعت پرس و جو تاثیری دارد نه داده ای مصرف میکند. فقط یک سطح دسترسی تعریف میشود و میگذاریم روی جدول و فقط بخشی از جدول را میبینیم. داخل view میتوانیم insert کنیم ولی واقعا داخل جدول اصلی هست البته هر چیزی هم نه مثل جدول پایه مثلا مقدار تکراری یا مقدار null روی کلید نمیتوانیم درجی که قوانین جامعیت را نقض نکند و فقط ستون های مورد نظر را میتوانیم ببینیم. حافظه مصرف نمیکند و تاثیری روی سرعت ندارد و روی جدول پایه قرار میگیرد. الزامی وجود ندارد جلوی view چیزی بگذاریم. یعنی کل ستون های جدول پایه میاد بعد میتوانیم شرط خودمان را بگذاریم. دقت کن برای تغییراتی مثل insert, update, delete باید grant به اعضا داده شود و اگر بخواهیم اجازه را از یکی بگیریم از دستور revoke استفاده میکنیم. دقت کن روی همان جدول view هم میتوانیم یک کوئری بزنیم. اگر بخشی از کلید null شود قانون جامعیت موجودیت نقض شده است. محتوای view، در جایی ذخیره نمیشود و از دید کاربر یک محتوای جدا ولی از دید سیستم همان جدول پایه هست که جدول استخراج شده است. دقت شود که view همانند جدول پایه باید کلید داشته باشد و کلید آن چند مقداری مرکب نباشد این میشود از نظر تئوری، و از نظر تئوری باید کلید داشته باشد دقت کن ویو خصوصیت جدول خودش را دارد. یعنی جامعیت درون رابطه ای را رعایت کرده باشد از نظر تئوری درست هست ولی هر چیزی که دلت بخواهد نمیتوانی داخلش بریزی چون از تئوری درست هست ولی از نظر عملی معلوم نیست و اگر یک جدول از نظر تئوری درست نباشد کلا درست نیست. کلید هم باید کلید جدول پایه باشد. تمام قوانین مدل رابطه ای برای view هم باید برقرار باشد چه از نظر تئوری چه از نظر عملی. مثلا میتوانی یک مقدار insert کنی ولی یک سری ستون ها را مقدار دهی نکنی و null شود از نظر تئوری مشکلی ندارد ولی از نظر عملی ممکن هست ستون های مورد نظر not null تعریف شده باشند و امکان اتفاق افتادن نباشد. دقت کن در جامعیت ارجاعی یکی از بخش کلید ها هم نباشد غلط هست. دقت کن کلید برای ما مهم هست نه سایر مقادیر insert شده. بنابراین insert کردن روی ویو امکان پذیر نیست چون بحث ما ذخیره سازی هست ولی از نظر خواندن مشکلی ندارد ویو. دقت شود که اگر group by بیاد پذیرا بودن ویو از نظر تئوری غلط هست یعنی مثلا مشخص نیست اصلا میانگین چه درسایی منظورمان هست و جدول اصلا ستون مجازی دارد. اگر در ویو به شرطی اینکه ستون مجازی نداشته باشیم مثل AVG میتوانیم روی 2 تا جدول باشد، کلید اینا هم باید اوکی باشد و در کل باید از نظر تئوری و عملی 1 جدول باشد و کلا درست نیست تا ببینیم از لحاظ عملی insert آن به چه شکلی هست. کلا ستون مجازی داشتیم مثال توابع آماری کلا قضیه کنسل هست.

فصل هفتم بخش سوم:

کلا هر چیزی که ستون مجازی مثل توابع آماری، group by و distinct ایجاد میکند نباید ویو داشته باشد. مثلا distinct اصلا کلید ندارد که بخواهد روش ویو بکشی هر جدولی که کلید نداشت ویو هم ندارد. Select درست هست که صاحب کلید باشد از جدول برای ویو. Cross join دقیقا یک عملگر هست و دقیقا ضرب دکارتی انجام میدهد، این دستورات DML هست. هم خود دستور cross join هم کاراکتر , میتواند برای cross join استفاده شود. اگر ستون مشترک داشت 2 برا میاد چون ضرب دکارتی هست. inner join ضرب دکارتی شرط دار هست. natural join یکبار ستون مشترک را میاورد. در join که همان equity join هست فقط از مساوی و برابری استفاده میشود نه از کوچکتر و بزرگتر همچنین ستون مشترک 2 بار میاد، و حالت خاصی از inner join هست. در جبر left outer join یا فرا پیوند در جبر همان فراتر از الحاق بود و یک سری موارد بهش اضافه میشود و ستون مشترک 1 بار میومد ولی اینجا فرق میکند تمام پیوند های پذیر در خروجی میاد بعلاوه پیوند ناپذیر چپ هم میاد و ستون مشترک هم 2 بار میاد در SQL اگر میخواهی 1 بار بیاد مشترک باید از natural left outer join استفاده کنی دقت کن natural نیست. اگر inner join بگی پیوند پذیر ها میاد و ستون مشترک 2 بار میاد ولی natural اضافه کنی 1 بار میاد. اگر inner join هیچ شرطی نداشته باشد میشود ضرب دکارتی یا همان cross join. دستورات امنیتی: revoke و grant: این دستورات به کاربران اجازه میدهند که در جداول insert کنند یا نکنند. یا همین برای delete و update و reference و یا select که همان خواندن هست. با revoke همین حق را پس میگیریم. Public بگذاریم جا نام میشود به کل کاربران از قبل تعریف شده و پسا تعریف شده. وقتی check میدهی باید حالت true باشد تا جواب بدهد. دقت کن not exist یعنی حالت true, false اگر جلوی آن خالی باشد میشود true و اگر پر باشد میشود false. Assertion میاد پسا انجام کار مثل insert, update, delete روی جدول چک میشود، اگر سطری اون قانون ما را رعایت نکند مخالفت میشود و این قانون جامعیت خارجی هست که توسط DBA یا طراح بانک طرح شده است. check جامعیت خارجی هست مثلا نمره 0 تا 20 باشد و برای چک کردن این از دستور check استفاده میکنیم تا این را کنترل کند.

فصل پنجم بخش اول:

حساب رابطه دامین و تاپلی وقتی با هم برابر هست که عبارت امن باشد و قدرت حساب رابطه ای دامین و تاپلی از جبر رابطه ای قوی تر هست. زیرا این 2 مربوط به فضای تحلیل هست یعنی ادراکی عام. تحلیل ادراکی عام و طراحی ادراکی خاص هست. از نظر توان رابطه ای با جبر رابطه ای برابر هست. قدرت بیان بیشتری دارد و اصلا عملگر نداریم تقریبا به زبان انسان هست و نیاز نیست جزئیات را بگوییم. حساب رابطه ای به زبان انسان نزدیک تر هست. کلا ترتیب نوشتن چه در SQL چه در جبر و چه در حساب رابطه ای: جدول، سطر و ستون. اگر حساب رابطه ای امن باشد میتوانیم توسط جبر رابطه ای بنویسیم. اینجوری بخون سطر هایی که متعلق به جدول مورد نظر هستند که ستون مورد نظر آنها شرط را دارا هست. برای قسمت سطر های شرط میگذاریم و برای پراجکت که همان ستون های مورد نظر ما هستند اطلاعاتی که سوال از ما میخواهد را قرار میدهیم مثلا میگوید مشخصات پس طبیعی هست که همه ستون های جدول را قرار بدهیم. مکان جستجو با تمام ستون میاد. سطر همان حکم where دارد اون اول هم خروجی هست. هر موقع خواستیم همه حالات یک جدول را بررسی بکنیم مثل تقسیم، از سور عمومی استفاده میکنیم. اگر یک ستون خاص را خواستیم، اولا باید ستون مورد نظر باشد و دارای آن شرط باشد، باید اون ستون دارای موارد سایر ستون ها باشد که با سور وجودی مشخص میکنیم. اگر خواستیم بخشی را نشان بدهیم این کار را بکنیم اگر کل را خواستیم دیگر بدیهی هست و نیازی نیست. دقت کن وقتی این موارد را میگذاریم همگی باید not null باشند.

میتوانی ستونی که روش شرط هست را با ستونی که در خروجی میخواهد منها کنی تا این موارد وابستگی بدست بیاد. دقت کن فارغ از هر چیزی برای جداول مورد نظر همه ستون ها را باید بنویسی. دقت کن الحاق باید روی چه ستونی صورت بگیرد؟ روی ستونی که از ما آن اطلاعات را میخواهند و همچنین دقت کن اون ستونی که نمیخواهیم الحاق شود و الحاق را از کار بیندازیم باید نام آن را در یکی از جداول متفاوت کنیم تا از کار بیفتد. پس از جدول اصلی میگیریم دستم میبریم اونجا یقه یک نفر را میگیرد بعد به همراه تخصص آن برمیگردیم. گرفتن دست یعنی کلید خارجی باید باشد. اگر رو هر دو تا ستون الحاق بگیریم نام رئیس هایی رو میدهد که داخل همان دانشگاه هایی که هستند و درس میدهند یکی باشد ولی ما این را نمیخواهیم چون میتواند طرف رئیس دانشکده ریاضی باشد ولی معارف درس بدهد. یعنی روسایی را میدهد که رئیس همان جایی هستند که درس میدهند. همه حالات یک اتفاق یعنی به تقسیم اشاره دارد اینجوری فکر کن به ازای دانشکده ای بعد همه دکتری داشته باشند یعنی دانشکده (سور وجودی) همه دکتری داشته باشند (سور عمومی). جدول ها را منها خروجی کنی ستون هایی که میخواهیم ستون های جلوی سور عمومی بدست میاد. جدول دوم را منهای خروجی بکنی کم و کسری هاش بدست میاد. اون ستون هایی که در شرط هستند باید زوری not null باشند. ستون مشترک در الحاق یکبار میاد. اگر در جدول دوم دیدی ستون مشترک نیومد بدان در جدول اول یکبار آمده است و از تکرار جلوگیری میشود.

قسمت پنجم بخش دوم:

حساب رابطه ای سطری یا تاپلی داریم، و یک متغیر سطری یا تاپلی داریم، در متغیر سطری هر سری یک سطر قرار میگیرد، وقتی داخل آن چیزی پیدا شود true و وقتی چیزی پیدا نشود false میشود. متغیر سطری باید دامنه آن تعریف شود یعنی بر اساس جدول s پس دیگر نمیتوانیم مقادیر جدول p را درون آن بریزیم. متغیر بازه ای هم گفته میشود. دقت کن اگر نام متغیر سطری SX باشد و ما سطر را درون آن بریزیم و بخواهیم دنبال مقداری خاصی بگردیم میگیم SX.city دقت کن. سور مقدار برنمیگرداند یا فالس هست یا ترو کلا میگوید بزرگتر از صفر هست یا نه. سور وجودی را با exist و سوری عمومی را با FORALL نمایش میدهند. for all میگوید برای همش همه باید این شرط را داشته باشند یعنی فقط یکی داشته باشه برعکس قبلی درست نیست.

فصل ششم:

هر چه تعداد شاخص های یک جدول بیشتر شود سرعت insert در آن جدول کمتر میشود. شاخص عوض کنی کاتالوگ هم تغییر میکند چون در کاتالوگ هست. شاخص میتواند روی چندین ستون باشد. روی بیشترین چیزی که جستجو میشود شاخص تعریف میکنیم. صفت ترکیبی داریم ولی صفت مرکب نداریم و مجاز نیست. گرافی که دور برداشته میگویند درخت. هر چه قدر فاکتور انشعاب بیشتر عمق کمتر میشود. در درخت روی سطوح حرکت میکنیم پس عمق کم بهتر هست چه وقتی عمر کمتر میشود؟ فاکتور انشعاب زیاد باشد. درجه 3 هست پس تعداد کلید میشود 2 تا t یعنی فاکتور انشعاب و تعداد کلید میشود t-1. سومی را وارد کنی درخت شکسته میشود دقت کن. B+ درخت متوازن هست یعنی ریشه تا برگ در تمام برگ ها اندازه یکسان دارند. SQL روی کلید اصلی همیشه cluster تعریف میکند و بقیه بدون cluster هستند. نرخ انتقال که میشود L/R یک پدیده ثابت هست این کاردینالیتی جدول هست که متغیر هست و زیادتر شود کند تر میشود. دو هدف سیستم های ذخیره و بازیابی را دنبال میکنند سرعت باید زیاد و زمان کم شود و حافظه هم کم شود. شاخص درختی عمق کم خوب هست. افزایش سرعت از طریق درخت پهن و کم عمق صورت میگیرد. به ستون هایی که شاخص در آنها تعریف میشود کلید جستجو گفته میشود. به جای نگهداری کل جدول فقط کلید جستجو را نگه میدارد و عرض کمتر میشود واسه همین هست که اون زمان شده است 1 از 6 . دقت کن سرعت جستجو در جدول پایه بیشتر میشود. همانطور که سرعت پرس و جو زیاد میشود ولی insert هم از اونطرف کند تر میشود. رکورد اول در جدول پایه وارد میکنیم بعد یک زمان اضافه ایم باید بگذاریم تا در جدول شاخص وارد شود. افزونگی یا تکنیکی هست یا طبیعی تکنیکی مثل تعریف شاخص یا کلید خارجی و طبیعی هم میشود همان محتوایی که در نرمال سازی صحبت کردیم. با نرمال سازی افزونگی محتوایی نداریم ولی دچار افزونگی تکنیکی میشویم بخاطر تعریف کلید ها. زیادی شاخص باعث کندی میشود. شاخص بخشی از متا دیتا هست و در کاتالوگ ذخیره میشود. ترکیبی همان چند ستون هست. شاخص میتواند نزولی و صعودی مرتب شود پس مرتب هست چه صعودی چه نزولی برای حذف از دستور drop index میکنیم که اگر حذف کنیم از کاتالوگ هم حذف میشود. اگر چیزی به چیزی وابسته باشد پاک نمیشود. 2 نوع شاخص داریم از نوع مرتب شده همینکه گفتیم و نوع دوم از نوع hash.

1. شاخص مرتب شده: ساختار آرایه و B+ tree میتواند برای این نوع شاخص استفاده شود که دومی بهتر هست چون از مرتبه log n در مبنای p هست، مثل اون کوئری هایی که گفتیم شاخص مرتب شده توسط این 2 تا هست. آرایه یک ساختمان داده خطی و درخت یک ساختمان داده غیر خطی هست. خود شاخص مرتب شده به 2 صورت هست: clustered index and no clustered index دقت کن ما فقط یک clustered index میتوانیم داشته باشیم که کلید اصلی همان هست و این شاخص سیستمی هست نه کاربردی در clustered index ترتیب منطقی و فیزیکی یکی هست یعنی اگر یک چیز از لحاظ منطقی اول هست از لحاظ هارد هم اول هست. و اگر دوم باشد در هارد هم دوم هست. ترتیب باید حفظ شود که توسط DBMS انجام میشود به این cluster شاخص اصلی یا اولیه هم گفته میشود. مقادیر شاخص مرتب شده clustered باید یکتا و منحصر به فرد و حتما به صورت صعودی باشند پس در ایجاد یک شاخص cluster حتما unique و ASC باید باشد تا سطر تکراری نداشته باشد این به صورت خودکار اجرا میشود. دقت کن not null هم لزوما نیست باشد چون وقتی کلید اصلی تعریف میکنی خود به خود خالی نمیماند توسط DBMS همچنین یکتا هم میشود. دقت کن روی همان ستونی که شاخص میگذاری پرس و جو سریعتر میشود نه روی پرس و جو سایر ستون ها. پرس و جو روی ستون شاخص دار مینویسی روی جدول شاخص یا جستجو کلید جستجو میکند یعنی روی اونکه زمانش 3 ثانیه طول کشیده است ولی پرس و جو سایر ستون ها که شاخص ندارد رو جدول پایه و اصلی حرکت میکند چون دیگر اون جدول جستجو کلید شاخص دار وجود ندارد. سطح به توان عمق داریم در سطح 1 یا عمق صفر میشود 10، سطح به توان عمق تعداد گره را مشخص میکند. تعداد سطح یکی از عمق بیشتر هست. از سطح یکی بریم پایین میرویم عمق 1 بعد یکی دیگه پایین برویم میشویم عمق 2. H همان سطح هست.