بنام خدا

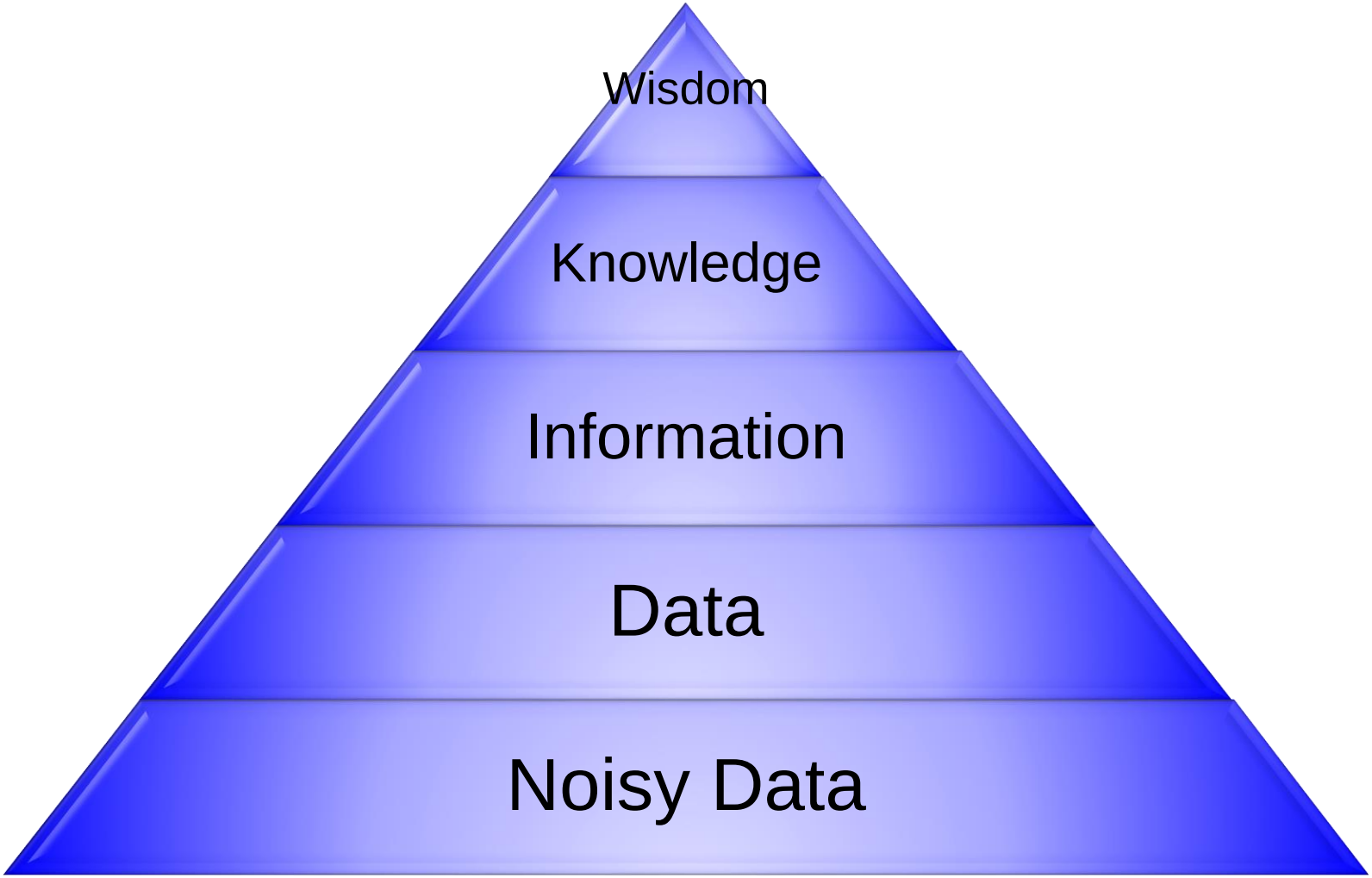
DATA WAREHOUSE

علیرضا بصیری

مراجع

- Han, Jiawei, Micheline Kamber, and Data Mining. "Concepts and techniques." *Morgan Kaufmann* 340 (2006): 94104-3205.
- Kimball, Raiph. *The data warehouse toolkit*. John Wiley & Sons, 2006.
- Inmon, William H. *Building the data warehouse*. John wiley & sons, 2005.

Data , Information & Knowledge



دیتا: یک داده خام است که اطلاعاتی به ما نمی دهد --> توضیح در مورد داده نمی دونم و فقط می دونیم یکسری از جدول ها است که داره پر میشه ولی هیچ واقعیتی به ما منتقل نمی کنه
Information: ولی بعد فهمیدیم که مثلا مشتری ما در تاریخ فلان ایجاد شده و این محصول رو از ما خریده و ینی این اطلاعات رو از این گزارش می فهمیم پس توی این لایه با Information سر و کار داریم --> داده رو وقتی فهمیدیم ینی Information برامون واضح است
noisy data: قبل از دیتا است که اگر چالشی داخل دیتا وجود داشته باشه باهاش سر و کار داریم - مثلا فرض بکنیم داخل داده ها پر از اشکال است و دیتای درستی نیست به این ضعف ها میگیم noisy data
Knowledge : مثال این برای فروشگاه --> مثلا اگر دوتا کالا کنار هم باشه خریدش می ره بالاتر پس چیزایی که بهم خرید میشن میشه دانش ینی پترن رفتیم بررسی کردیم - مثلا رفتیم به این دانش رسیدیم که توی 10 روز اخر تابستون بیشترین فروش لباس پاییزه رو داشتیم ینی یک تکرار داخلش دیدیم حالا میتونیم ازاین استفاده بکنیم یا نکنیم
wisdom: این عملا خرد است --> اگر این دانش ها رو کشف کردیم و در عمل هم داخل سازمان به کار گرفتیم این وقت این شرکت ما یک شرکت با خرد است ینی به کار گیری این دانش در امور شرکت

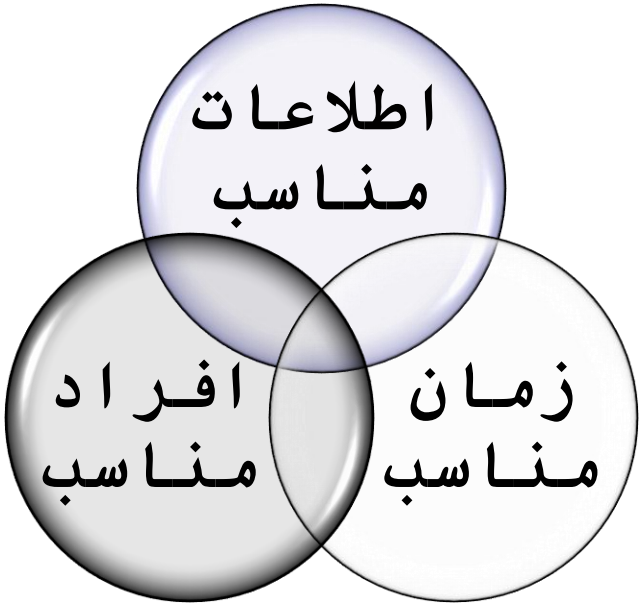
هوش تجاری(Business Intelligence) چیست؟

- جمع‌آوری، نگهداری، تجزیه و تحلیل و انتشار داده/اطلاعات به منظور تسهیل در تصمیم‌گیری‌های مدیریتی.

توی این درس میخوایم به این برسیم:
یک بستر داده ای می خوایم فراهم بکنیم برای هوشمندی یک سازمان - برای تسهیل درتصمیم گیری های لایه بالای سازمان که این میشه هوش تجاری که دنبال چیه؟ جمع اوری و نگهداری و تجزیه و تحلیل و انتشار داده ها به منظور این که بخوایم این کار رو انجام بدیم

توی دیبی 1 می خواستیم تراکنش ها رو ثبت کنیم ولی اینجا می خوایم از اون داده های که ثبت شده استفاده بکنیم که این کارها رو انجام بده --> تصمیم های لایه های مدیریتی سازمان رو انجام بده

هوش تجاری



هوش تجاری یعنی دراختیار قراردادن **اطلاعات مناسب** به **افراد مناسب** در **زمان مناسب** برای اخذ **تصمیم مناسب**

فرد مناسب: ینی به هرکسی نمی خوایم دیتا و اطلاعات رو نشون بدیم و این فردی که مد نظره فردی است که میخواد تصمیم گیری انجام بده پس لایه عملیاتی سازمان نیست ینی مخاطب ما اصلا نیست و میخوایم اطلاعات درست رو بهش نشون بدیم و اطلاعات رو میخوایم خیلی سریع بهش نشون بدیم

پس تصمیمات لایه بالا باید توی زمان مناسب انجام بشه پس توی این تعریف داریم می بینیم که زمان در اختیار قرار دادن داده بسیار برامون مهم است ینی اون فرد داده رو باید سریع و راحت بگیره

نکته: هوش تجاری بدرد هر سازمانی نمی خوره

نکته: ریکامندر سیستم : هدف اینه که ایتمی رو به یک مشتری پیشنهاد بکنیم که حدس می زنیم خوشش بیاد و یا ایتمه تکرار پذیره و می خواد دوباره بخرتش یا تکرار پذیر نیست --> کلا چیز خوبی هست ولی پیاده سازیش داستان داره

اهمیت هوش تجاری

Top Tech Investment in 2019

Rank	Government Priorities	% Respondents
1	**BI/data analytics**	**43%**
1	**Cyber/information security**	**43%**
3	Cloud services/solutions	39%
4	Core system improvements/transformation	33%
5	Software development/upgrades	26%
6	Infrastructure/data center	23%
7	AI/machine learning	22%
8	Technology integration	21%
9	Customer/user experience	20%
10	Mobile applications	19%

Gartner's 2019 CIO Agenda Survey gathered data from 3,102 CIO respondents in 89 countries and across major industries, including 528 government CIOs. Government respondents are segmented into national or federal; state or province (regional); local; and defense and intelligence, to identify trends specific to each tier.

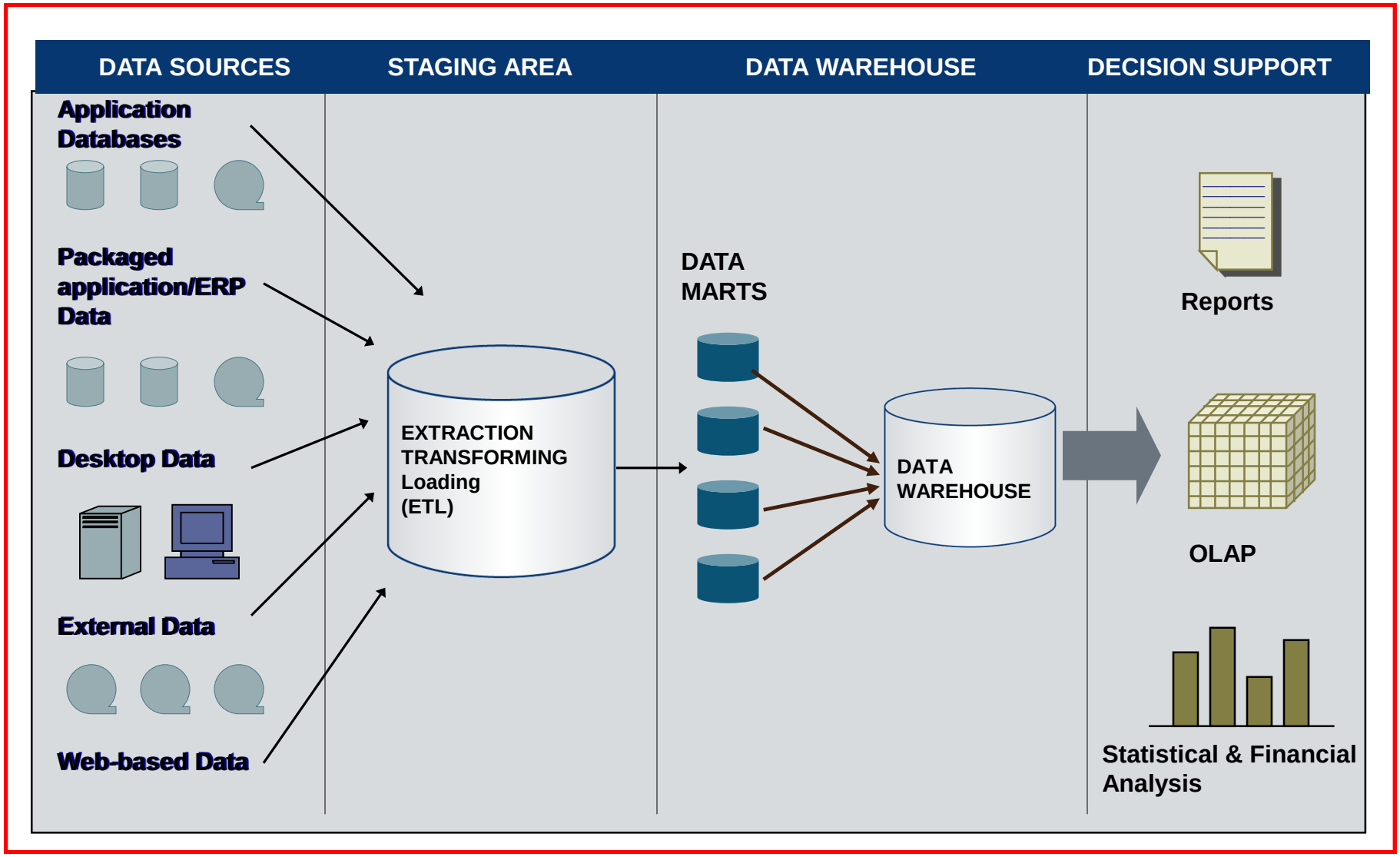
ابزارهای هوش تجاری

- BizzScore Suite
- Board Management IntelligenceToolkit
- Business Objects Enterprise
- IBM Cognos
- JasperSoft
- Microsoft BI tools
- Microstrategy

- Oracle
- WebFocus
- Tableau Software
- Style Intelligence
- SAS
- SAP
- QlikView
- Pentaho BI Suite
- Actuate
- .....

معماری هوش تجاری

BI



می خواستیم برای تصمیم گیری به مدیران یک سازمان توی زمان مناسبی جواب مناسبی بهشون برگردونیم --> اگر یک سازمان منابع داده ای مختلفی داشته باشد ینی پایگاه داده های مختلفی داشته باشه اینا میشن بدنه سازمان که دارن امور روزمره یک سازمانی رو انجام میدن
اگر مدیر عامل یک شرکت بخواد تصمیم گیری بکنه ایا فقط به یه دونه از این منابع داده ای نیاز داره؟ نه --> بلکه یک نگاه جامع میخواد
هوش تجاری میگه برای ما بستر داده ای رو سعی میکنه فراهم کنه که ما توی اسرع وقت برسیم به اون فیلدهایی که میخوایم --> پس بستر شکل روبه رو رو باید فراهم بکنیم
توی سورس: اپلیکیشن های مختلفی داریم - سازمان های مختلفی داریم
فرض میکنیم ما رفتیم داخل بیمارستان منابع داده ای مختلفی داره و هماهنگ کردیم با بخش های مختلف که این منابع رو بگیریم و می خوایم یک فرایندی بنویسیم که داده ها از این منابع داده ای مختلف اون چیزی که ما میخوایم و برای سازمان مهمه رو توی یک فضایی به اسم DATA WAREHOUSE بیاد ذخیره بکنه --> پس میخوایم منابع داده ای رو یکجا به اسم انبار داده تجمیع بکنیم برای اینکه بخوایم این کا رو بکنیم --> نکته ای که هست اینه که نمیشه دیتا رو یک راست ریخت توی DATA WAREHOUSE چون معمولا تایپ هاشون با هم فرق داره یا با هم نمی خونن یا چالش هایی دارند یا.. پس از یک فضایی استفاده میشه به اسم STAGING AREA که این STAGING AREA یک فضایی انگار temp است --> این داده ها رو معمولا منتقل میکنیم به این فضای موقتمون و اگر دیتا ناسازگار بود یا .. اینجا دیتا رو تمیز و یکپارچه می کنیم پس این بخش فضای موقتی است قبل از انبار داده و دیتایی که اینجا اومده با فرایند ETL خونده شده که اومده اینجا الان، بعد از این STAGING AREA باید بیارمش حالا توی انبار داده
مدل استار --> هدف اصلی این است که بسیار جوین ها کم میشه : فضای انبار داده

ادامه 2:
لایه بعدی چیه؟ می تونیم بیایم از خود همین انبار داده ای که فراهم کردیم کارها رو انجام بدیم مثل گزارش گیری یا OLAP یا ...
توی انبار داده دیتای سنگینی وجود داره
دیتا مارت: یک افراز منطقی از انبار داده است پس فیزیکالی جدا نیست داخل اینجا، بلکه یک جا است دیتای ما --> توی DATA WAREHOUSE هم با جدول سر و کار داریم و این جدول ها با توجه به این که این جدول توی کدوم بخش سازمان می خواد استفاده بشه یک لیبل بهش می زنیم مثلا داخل دانشگاه بخش اموزش ما میایم توی انبار داده اون جدول هایی که مربوط به تصمیم گیری در مورد واحد اموزش است رو براشون یک لیبل می زنیم اول این جدول ها --> پس داخل دانشگاه می تونیم دیتا مارت برای واحد اموزش داشته باشیم و دیتا مارت برای واحد دانشجویی و...

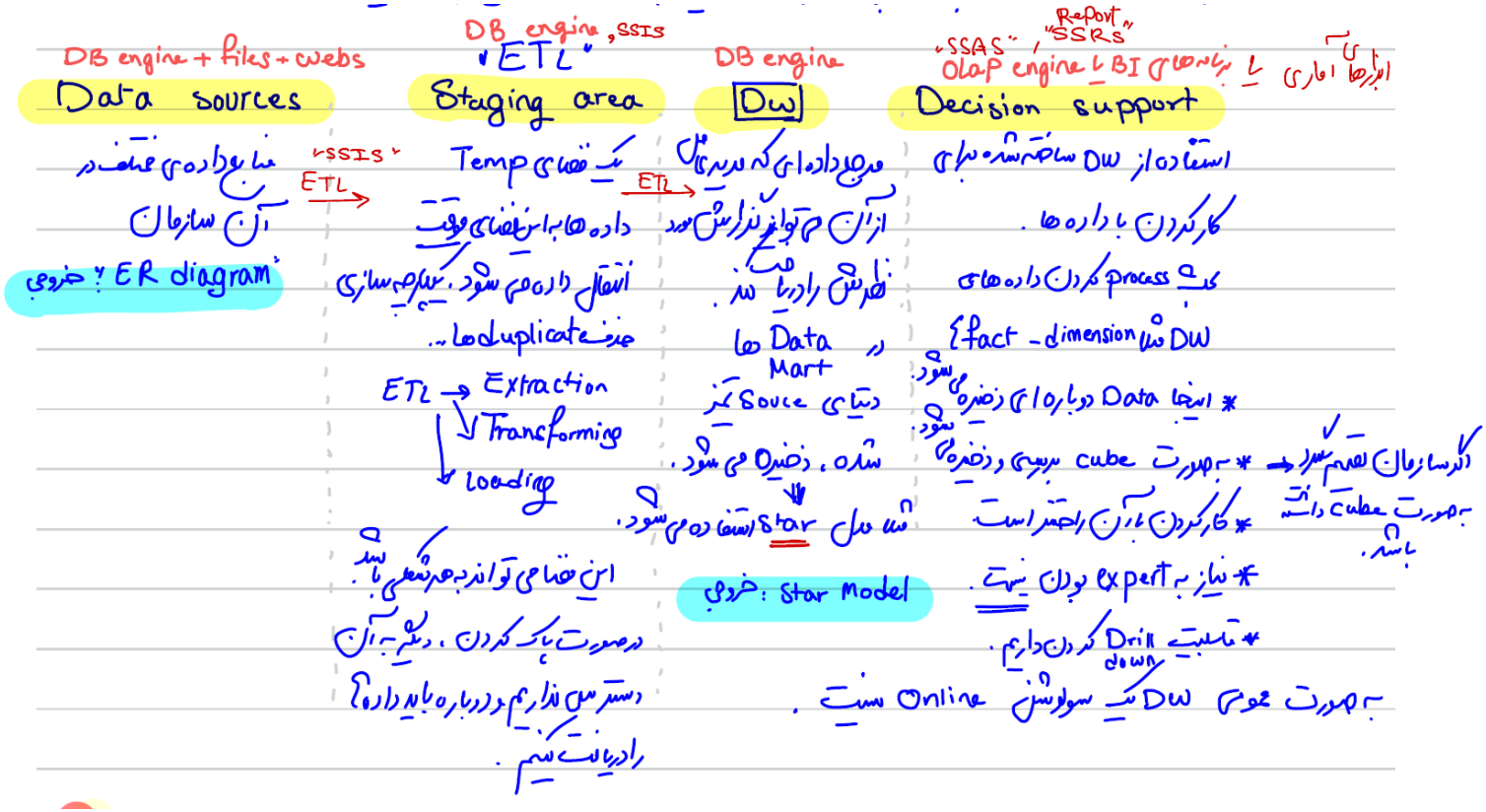
نکته: انبار داده یک راه حل انلاین به صورت عمومی نیست چون براساس گذشته داره تصمیم میگیره --> تصمیم گیری توی سطح بالای سازمان است برای همین افلاین است - معمولا توی یک دوره زمانی که عرفش یک روزه است این انبار داده به روز میشه
ETL یک پروسه افلاینی است

یکی از روش هایی که قراره اطلاعات رو بخونیم از سورس داده ای و بیارمیش توی انبار داده یا اینکه توی طراحی متفاوتی ذخیره اش بکنیم اینه که پروسیجر بنویسیم برای STAGING AREA توی STAGING AREA می تونیم از ابزارهایی استفاده بکنیم حتما لازم نیست که از پروسیجر استفاده بکنیم اینجا مثلا از اوراکل استفاده بکنیم پس توی STAGING AREA میتونیم از ابزارها استفاده بکنیم که همون کارهایی که می خوایم به شکل پروسیجر انجام بدیم رو برای ما راحتر میکنه یا اینکه می تونیم از خود پروسیجرها استفاده بکنیم

ادامه 3:
نکته: olap engine میاد cube رو ساپورت می کنه

تکنولوژی های استفاده شده در هر لایه یا پیاده سازی ها:
توی سورس:
وب سرور - فایل - DB engine
: STAGING AREA
DB engine
توی DW:
DB engine
توی DECISION SUPPORT:
ابزارهای اماری یا برنامه های BI یا olap engine

ETL ها کجا قرار میگیرند؟ یکی از سورس می خونه و توی STAGING AREA لودش می کنه و یکی هم از این STAGING AREA که می خواد بخونه و انبار داده رو پر بکنه



برای ابزار ETL هم SSIS رو داره
ETL هایی که داره انجام میشه یک فرایند سنگین است
برای بحث رپورت DECISION SUPPORT ما SSRS داره

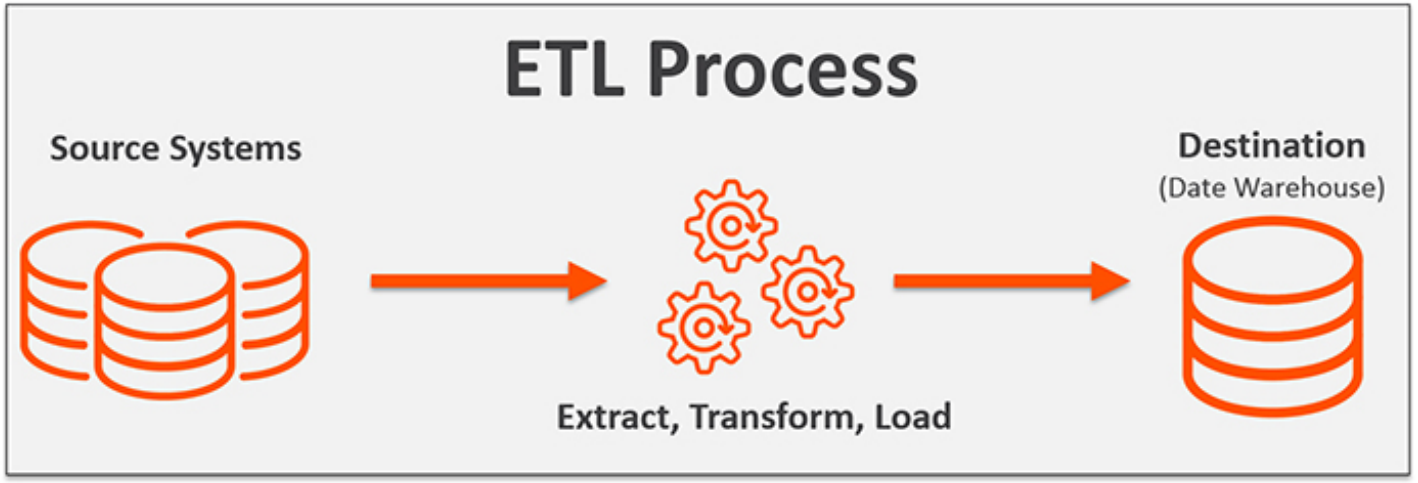
نکته: انبار داده یا BI یک راه حل برای سازمان است
یکی از مواری که به ما می تونه نشون بده یک چیزی از جنس انبار داده است یا نیست همین کاربرانش است --> وقتی که می ریم سراغ انبار داده تعداد کاربران به شدت کم است --> هدف تصمیم گیری است

نکته: انبار داده بسترش برای استفاده end user نیست و برای لایه بالای سازمان است که میخواد گزارش کلی بگیره از سازمان

توی کجا یک بک اپ باز از دیتا داشتیم؟ و دیتا بود باز؟ انبار داده؟

Extraction, Transformation, and Loading (ETL)

- استخراج داده ها از منابع داده ای مختلف، Extract
- تحلیل و تبدیل داده‌ها مطابق با نیازمندی‌های کسب و کار، Transform
- بارگذاری اطلاعات Loading



ETL ینی ما یک یا چندتا سورس اطلاعاتی داریم و می خوایم اطلاعاتش رو بخونیم و یه جا بریزیمش
برای اینکه بخوایم این کار رو انجام بدیم --> میشه با پروسیجر این کار رو انجام داد

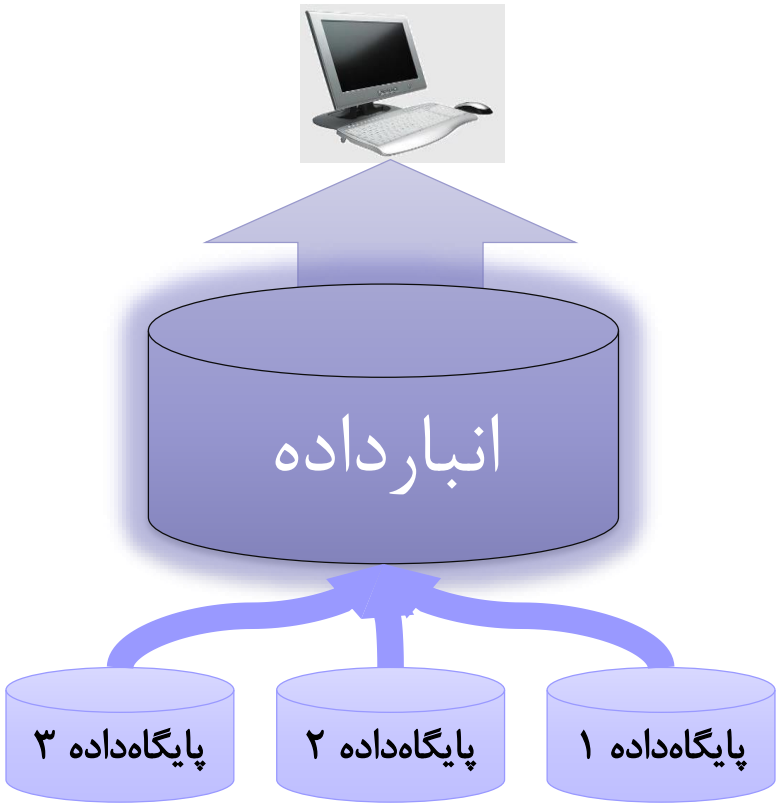
انبار داده افلاین است

مثلا 5 تا پروسیجر داریم و این 5 تا پروسیجر قراره هر شب به روز بشه پس این 5 تا پروسیجر ما میشه فرایند ETL --> یک ترتیب هم باید براش قائل بشیم (که اینم توی یک پروسیجر دیگه گفتیم که داخلش ترتیب اون 5 تا پروسیجر چجوری باشه)

Transform: حذف duplicateها (union, distinct , group by) و تبدیل داده ها وقتی که جنس و نوع آن ها متفاوت است (استفاده از case when) مثلا خانم ها رو گفته صفر و اقایون رو گفته یک توی سورس بعد داخل این STAGING AREA می خوایم تبدیلشون بکنیم به خانم و اقا پس از case when استفاده میکنیم

نکته: توی شرکت ها کسی که طراحی انبار داده رو انجام میده با کسی که ETL رو می نویسه متفاوت است

انبار داده (Data Warehouse)



انبار داده مجموعه‌ای از داده‌هاست که از منابع مختلف اطلاعاتی سازمان جمع آوری، دسته‌بندی و ذخیره شده و جهت انجام عملیات گزارش‌گیری و آنالیز در دسترس مدیران می باشد.

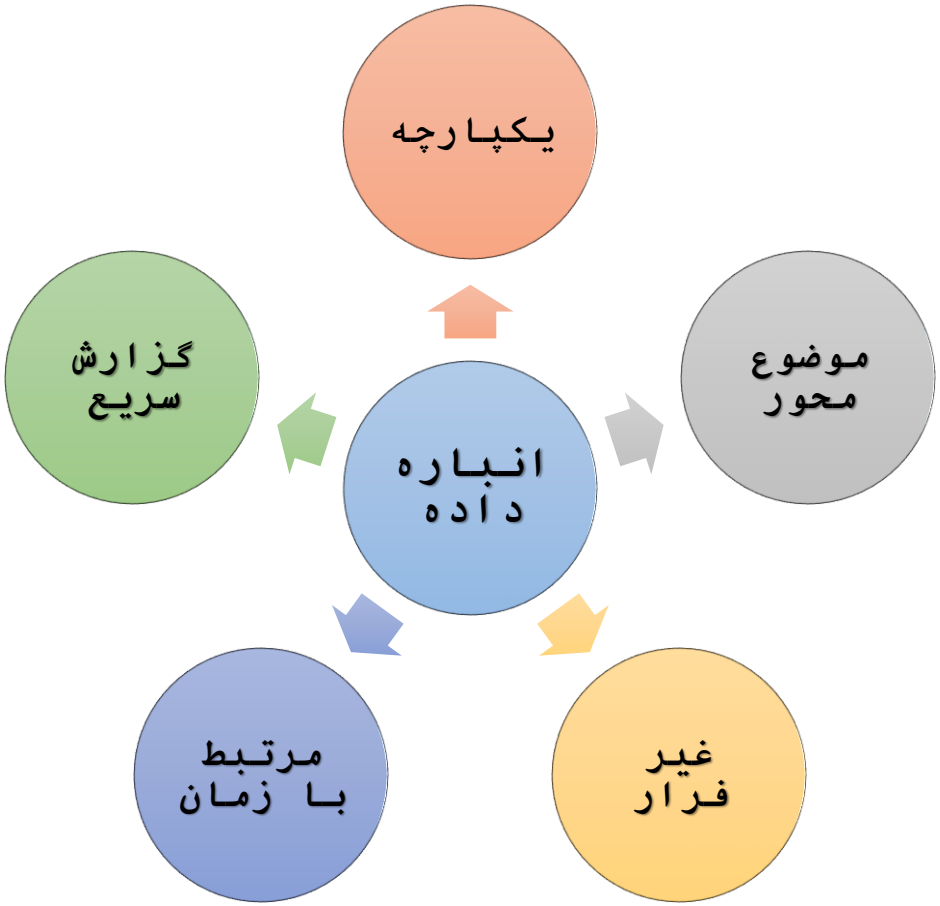
چرا به انبار داده و هوش تجاری نیاز داریم؟

1. نیاز به یک چارچوب تصمیم‌گیری برای تسریع در اتخاذ تصمیمات موثر
2. **دسترسی سریع به تمام اطلاعات موجود**
3. **امکان ساخت هر گونه گزارش به صورت دینامیک**
4. **امکان تحلیل اطلاعات به صورت Historical(مبتنی بر زمان) از کل به جزء(Drill Down) با استفاده از تجمیع ها(Aggregation)**
5. **امکان انجام تحلیلهای آماری و مبتنی بر داده کاوی**
6. **ایجاد مرجع واحد آمار و گزارشات**
7. **امکان برنامه نویسی و تولید نرم افزارهای خاص منظوره**

1: ینی یک بستری براش فراهم میکنیم که گزارشی که میخواد رو بتونه خیلی سریع ایجاد بکنه و بهش برسه
3: تمامی گزارش هایی که یک نفر می خواد رو براش جنریت بکنیم مثلا یک سامانه ای است که کسی نمی تونه ازش گزارش بگیره و برای گزارش گرفتن باید بره سمت شرکت تولید کننده سامانه و ممکنه خیلی دیر به دستش برسه پس باید این امکان وجود داشته باشه که خود اون فرد بتونه از اون سامانه گزارش بگیره ینی یک ابزار گزارش سازی ساده در اختیار اون مدیره هست
4: عموما یکی از ابعاد مهم انبار داده از بعد زمان است یا تاریخ

نکته: لزوما همیشه حجم داده ما زیاد نیست ینی با این دیفالت نریم جلو که هر جا که داده اش خیلی زیاده انبار داده داشته باشیم واقعا اینجوری نیست ینی حتی ممکنه ما یه جایی تصمیم بگیریم که انبار دادمون ویو باشه همه چیش --> اگر به جای جدول توی انبار داده یکسری ویو داشته باشیم چه مزیتی داره؟ ETL دیگه نداریم و حذف میشه

ویژگی های انباره داده (Data Warehouse)



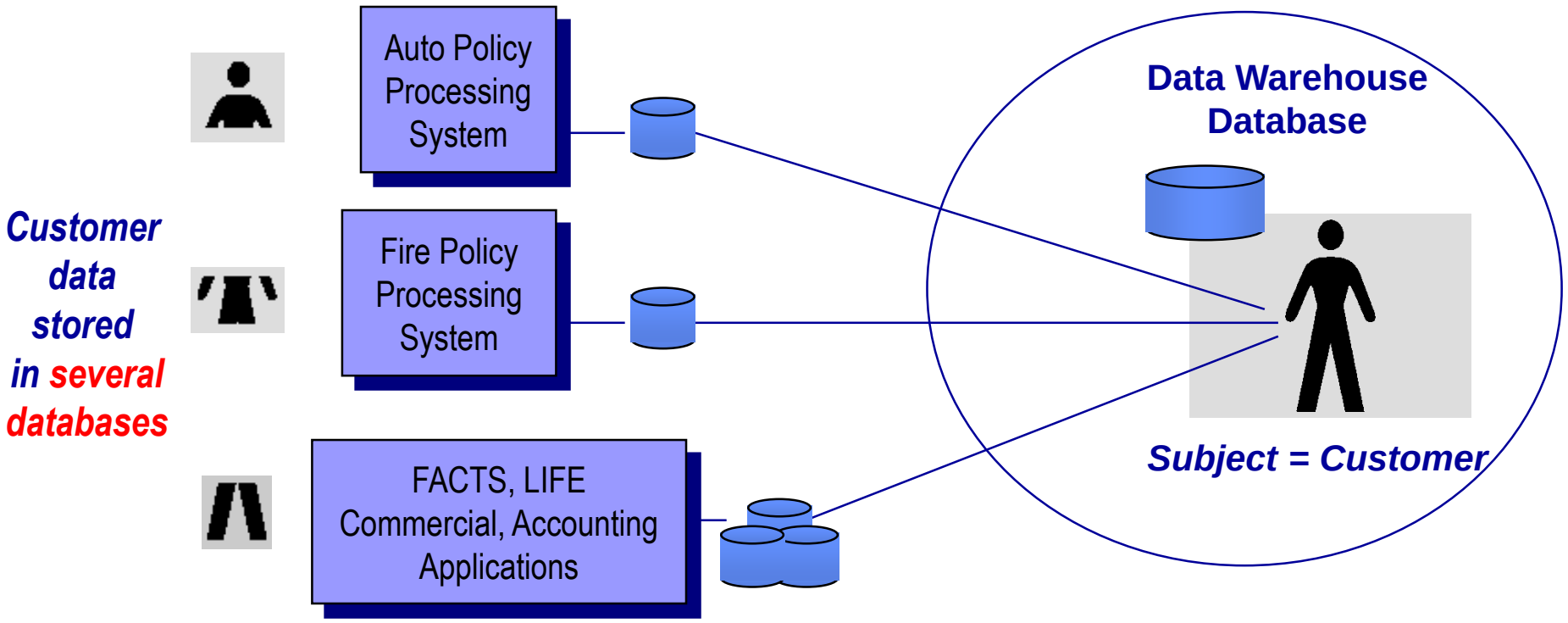
یکپارچه:
ما نمی خوایم چند انبار داده داشته باشیم برای یک سازمان بلکه میخوایم یک انبار داده داشته باشیم
--> اگر چند انبار داده داشته باشیم نزدیک سورس می شیم
پس همه تلاش رو میکنیم که یک مرجع داشته باشیم --> پس یکی از نشونه های انبار داده این است که یک مرجع داده ای ثابت داشته باشیم
موضوع محور:
خیلی نزدیک است به بیزینس - کسی که در زمینه IT با تجربه نباشد از انبار داده سر در نمی اورد
ینی این که سابجکت های مهم بیزینس داخل انبار داده ذخیره می شوند و جزئیات پایگاه داده های اپرشنال داخل پایگاه داده انبار داده نمیاد
غیر فرار:
به شکل جنرال داخل انبار داده insert داریم و select هم انجام میدیم ولی چیزایی از جنس update , delete نداریم
delete در اینجا ینی دیتایی رو تا دیروز داشتیم ولی امروز نمی خوایم داشته باشیم بلکه به این معنا نیست که بخوایم دیتایی رو حذف کنیم ینی اگر دیتایی توی انبار داده اومد همیشه هست
update نمیکنیم چون عملیاتی از جنس اپدیت کردن ربطی به انبار داده نداره
پس در حالت کلی یک پایگاه داده اگر insert اش رو بذاریم کنار read-only است و کسی قرار نیست write بکنه و write رو پروسه ETL هندل میکنه و پروسه ETL دیتا رو اپدیت میکنه
تمام تغییراتی که روی جدول های انبار داده اتفاق می افته به واسطه پروسه های ETL است

مرتبط با زمان:
بحث زمان و تاریخچه و هیستوری به عنوان یکی از ابعاد مهم حتما داخل انبار داده داریم
KPI چیه؟
نکته: مثلا شغل مشتری عوض شده و این شغل مشتری رو میخوایم تغییراتشو نگه داریم توی انبار داده و تا الان جایی نگه داشته نشده پس اخریش رو خوندیم و توی بعد مشتری توی بخش شغل نگهش داشتیم و از این به بعد می گن اگه تغییری کرد ما بیایم نگهش داریم ایا میشه؟ بله میشه --> توی DW اون چیزایی که قبلا بوده رو داریم الان و اون چیزی که جدید اومده رو اضافه میکنیم بهش --> همه تغییرات رو میتونیم نگه داریم؟ ایا این روش باگی داره؟ اره داره بسته به اون زمانی که می ذاریم به روز بشه اگر توی اون زمان بیشتر از یک تغییر انجام بشه ما دیگه اینو نمی تونیم هندل بکنیم (چون ETL یکبار فراخوانی میشه توی یک دوره و بعد می ره از سورس می خونه همون موقع هم)

گزارش سریع

Integrated(یکپارچه)

- **Data is stored once in a single integrated location (e.g. insurance company)**



-
سورس های مختلفی داریم و این دیتابیس های سورس همشون باید یکپارچه بشن

Data Warehouse—Time Variant(مرتبط با زمان)

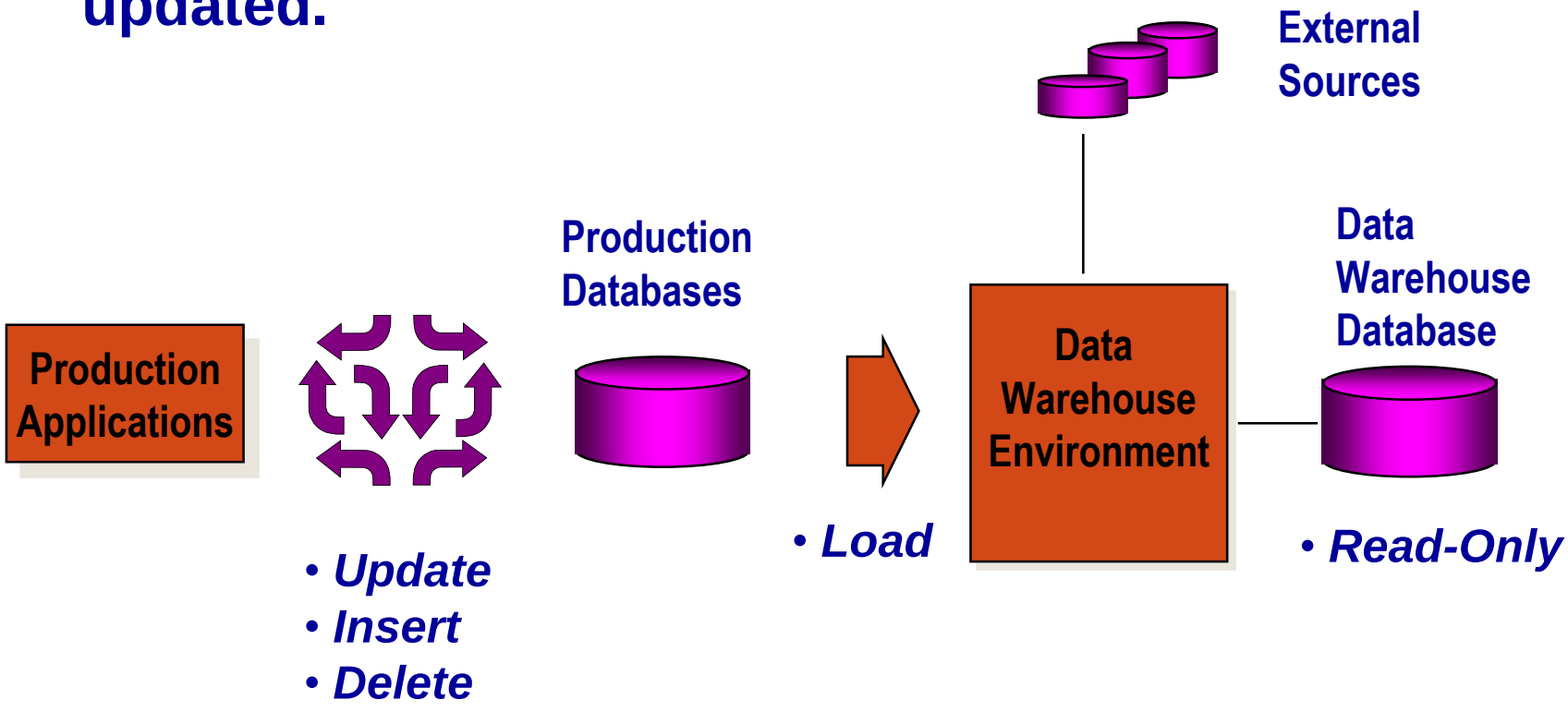
- The time horizon for the data warehouse is significantly longer than that of operational systems
  - Operational database: current value data
  - Data warehouse data: provide information from a historical perspective (e.g., past 5-10 years)
- Every key structure in the data warehouse
  - **Contains an element of time**, explicitly or implicitly
  - But the key of operational data may or may not contain "time element"

-
نکته: انبار داده یک المنتی از زمان حتما توش هست حالا یا به شکل ضمنی یا به شکل صریح

اطلاعات رو از جنبه زمان معمولا بین 5 تا 10 سال نگهش میداریم چرا این بازه رو مطرح میکنه؟ ممکنه یه جایی سخت باشه --> وقتی که مشتری عادت کرد به حضور این دیتا توی بازه 5 تا 10 ساله بعد سخت دل می کنه ازش - بحث های مالی اینا هم می تونه مهم باشه که نگهداریش دیگه سخت شده یا... - از طرف دیگه ای هم بحث های تصمیم گیری توی بازه 5 تا 10 ساله معنی داره

Non-Volatile (غير فرار)

- **Existing data in the warehouse is not overwritten or updated.**



-
با ETL تغییر میدیم insert , select رو
مکانیزم روتین هم فقط insert توسط ETL هست
از جنبه کاربر ما read-only می بینیمش--> کاربر فقط داره استفاده میکنه از داده ها ینی select داره می زنه

Data Warehouse—Subject-Oriented(موضوع محور)

- Organized around major subjects, such as customer, product, sales
- Focusing on the modeling and analysis of data for decision makers, not on daily operations or transaction processing
- Provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process

-
سابجکت های مهم مثلا توی حوزه فروش --> customer,product, sales هست که اینارو نگه میداریم و در نظر میگیریم

Data Warehouse vs. Operational DBMS

- OLTP (on-line transaction processing)
  - Major task of traditional relational DBMS
  - Day-to-day operations: purchasing, inventory, banking, manufacturing, payroll, registration, accounting, etc.
- OLAP (on-line analytical processing)
  - Major task of data warehouse system
  - Data analysis and decision making
- Distinct features (OLTP vs. OLAP):
  - User and system orientation: customer vs. market
  - Data contents: current, detailed vs. historical, consolidated
  - Database design: ER + application vs. star + subject
  - View: current, local vs. evolutionary, integrated
  - Access patterns: update vs. read-only but complex queries

-
OLTP مربوط به پایگاه داده اپرشنال است: دیبی 1
OLAP: ما کارهایی می خوایم انجام بدیم که روی حجم بالاتر از داده بتونه رقم بخوره و میخوایم دیتا رو انالیز بکنیم --> نیازمندی اینجا فرق میکنه
تفاوت:
OLTP رو نماینده DBMS اپرشنال ببینیم و OLAP رو نماینده دیتابیس انبار داده

OLTP vs. OLAP

	OLTP	**OLAP**
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented محور
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage کاربرد	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	GB to high-order GB	>=TB
metric	transaction throughput	query throughput, response

1: جنس یوزری که از oltp استفاده میکنه یک IT man هست ینی کسی که دیتابیس بلده و اون میاد از این استفاده میکنه و کار میکنه --> کاربر حرفه ای است ولی کاربری که از olap استفاده میکنه کاربر حرفه ای نیست
2: از جنس function --> ما توی oltp میخوایم عملیات روزانه رو هندل بکنیم ولی توی olap میخوایم تصمیم گیری های اون افرادی که سطح بالای سازمان هستن رو فراهم بکنیم
4: دیتا توی oltp دیتاهای روزمره و اپدیت هایی که میخوایم انجام بدیم و اطلاعات ریز است ولی توی olap دیتاهای historical و خلاصه شده و چند بعدی و یکپارچه شده است
5: usage توی oltp اینه که همیشه درگیر است ولی olap هر موقع که میخوایم تصمیم بگیریم استفاده میشه
6: access توی oltp هم رید است و هم رایت و ایندکس هم داره بخاطر اینکه میخواد اطلاعات فلان مشتری رو به روز بکنه و ایندکس به ما خیلی سریع کمک میکنه که مشتری پیدا بشه و اپدیتش رو انجام میده و می ره جلو مثلا اینجا نمره یک دانشجو مهم است ولی olap اینطور نیست مثلا نمره کلاس مطرح است یا نمره دانشگاه که اینجا حداقل مثلا 50 تا رکورد درگیر میشه
7: تعداد واحد کاری یا unit of work که داخل oltp انجام میشه خیلی کوچیک است ولی توی olap خیلی بالا است
8: records accessed یا رکوردهایی که توی oltp درگیر میشه چندتا رکورد بیشتر نیست ولی توی olap تعداد رکوردها خیلی بیشتر است
9: تعداد کاربرانی که از olap استفاده می کنند خیلی کمتر از oltp است : users
10: DB size یا حجم پایگاه داده توی oltp خیلی کمتر از olap است چرا olap زیاده؟ چون اینجا داریم دی نرمال طراحی میکنیم ولی توی oltp نرمال سازی شده
11: متریک: توی oltp میشه توان عملیاتی تراکنش و توی olap میشه توان عملیاتی کوئری و پاسخ

Why Separate Data Warehouse?

- High performance for both systems
  - DBMS— tuned for OLTP: access methods, indexing, concurrency control, recovery
  - Warehouse—tuned for OLAP: complex OLAP queries, multidimensional view, consolidation
- Different functions and different data:
  - missing data: Decision support requires historical data which operational DBs do not typically maintain
  - data consolidation: DS requires consolidation (aggregation, summarization) of data from heterogeneous sources
  - data quality: different sources typically use inconsistent data representations, codes and formats which have to be reconciled
- Note: There are more and more systems which perform OLAP analysis directly on relational databases

-
چرا انبار داده داشته باشیم؟
نکته: خیلی ها از سیستم ها از همون oltp استفاده می کنند و olap engine شون رو دارن روی oltp سوار می کنند --> انگار این DW رو حذف می کنند
مهمترین عاملی که ما رو می تونه به این سمت سوق بده که انبار داده رو به عنوان یک دیتابیس که کلی جدول داخلش هست و هر روز داریم پر می کنیم ایجاد نکنیم و اون رو به صورت یک ویو ببینیم حجم داده است اونم حجم داده کم
چالش هایی که وجود داره که انبار داده می تونه به ما کمک بکنه:
1- missing data
2- data consolidation
3- data quality : مثلا دیتامون ناسازگار باشه و ما روی داده های مختلف که داریم تجمیع میکنیم برامون ممکنه چالش برانگیز باشه

تاکید بر اینه که خیلی جاها نیاز داریم که انبار داده رو داشته باشیم

چرا از DW استفاده کنیم؟

- Quality داده ها چالش برانگیز باشد یا تغییراتی خواهیم انجام بدهیم که بیهوده باشد. و با بد Query توان هندل کرد.
- اگر Source ها زیاد باشد، باز هم بهتر است از DW استفاده کنیم. و عدم یکپارچگی اهمیت داشته باشد.
- نیاز به گزارش سریعتر باشد.
- اگر حجم داده ای که دریافت می شود هم باشد. لزومی به استفاده از DW نیست و معمولا بر روی Source ها گزارش می نویسند. table

---

From Tables and Spreadsheets to Data Cubes

- A data warehouse is based on a multidimensional data model which views data in the form of a data cube
- A data cube, such as sales, allows data to be modeled and viewed in multiple dimensions
  - Dimension tables, such as item (item_name, brand, type), or time(day, week, month, quarter, year)
  - Fact table contains measures (such as dollars_sold) and keys to each of the related dimension tables
- In data warehousing literature, an n-D base cube is called a base cuboid. The top most 0-D cuboid, which holds the highest-level of summarization, is called the apex cuboid. The lattice of cuboids forms a data cube.

-

: multidimensional

هدف اینه که ما بیایم یک مژری رو ک برامون مهم است از ابعاد مختلف ینی dimensions های مختلف بتونیم بررسی کنیم

dimensions: اون ابعادی که توی اون بیزینس مطرح است --> اون منظری که میخوایم مژره رو بشکنیم و ببینیمش --> یک نگاه است که می خوایم این KPI از این منظر هم ببینیم ینی مثلا می خوایم از منظر دانشجو نمره ها رو ببینیم یا از منظر دانشکده یا ... که این میشه ابعاد ما مثلا میخوایم مقدار فروش رو از جنبه های مختلف ببینیم مثلا از جنبه زمانی می خوایم ببینیمش ینی از یک نگاه مقداری که این فروشگاه فروخته می خوایم ببینیم توی هر ماه چقدر فروخته و توی هر سال چقدر فروخته و.. پس این مبلغ فروش رو فرض میکنیم که یک مژری است که میخوایم بررسیش بکنیم --> همش بعد زمان است ولی این بعد زمان یکسری فیچرهایی داره که به ما این قابلیت رو میده که میخوایم روز رو ببینیم یا ماه رو ببینیم یا سال رو ببینیم یا .. این یک بعد است یا یک dimension است

-

Multidimentional model : یک KPI از چند بعد توانند بررسی شود. ← یکی از روش‌های ذخیره اطلاعات انبار داده است.

(measure) شاخص‌ها

← توان از منظرهای مختلف، داده‌ها را مشاهده کرد.

← با استفاده از Drill Down کردن data cube

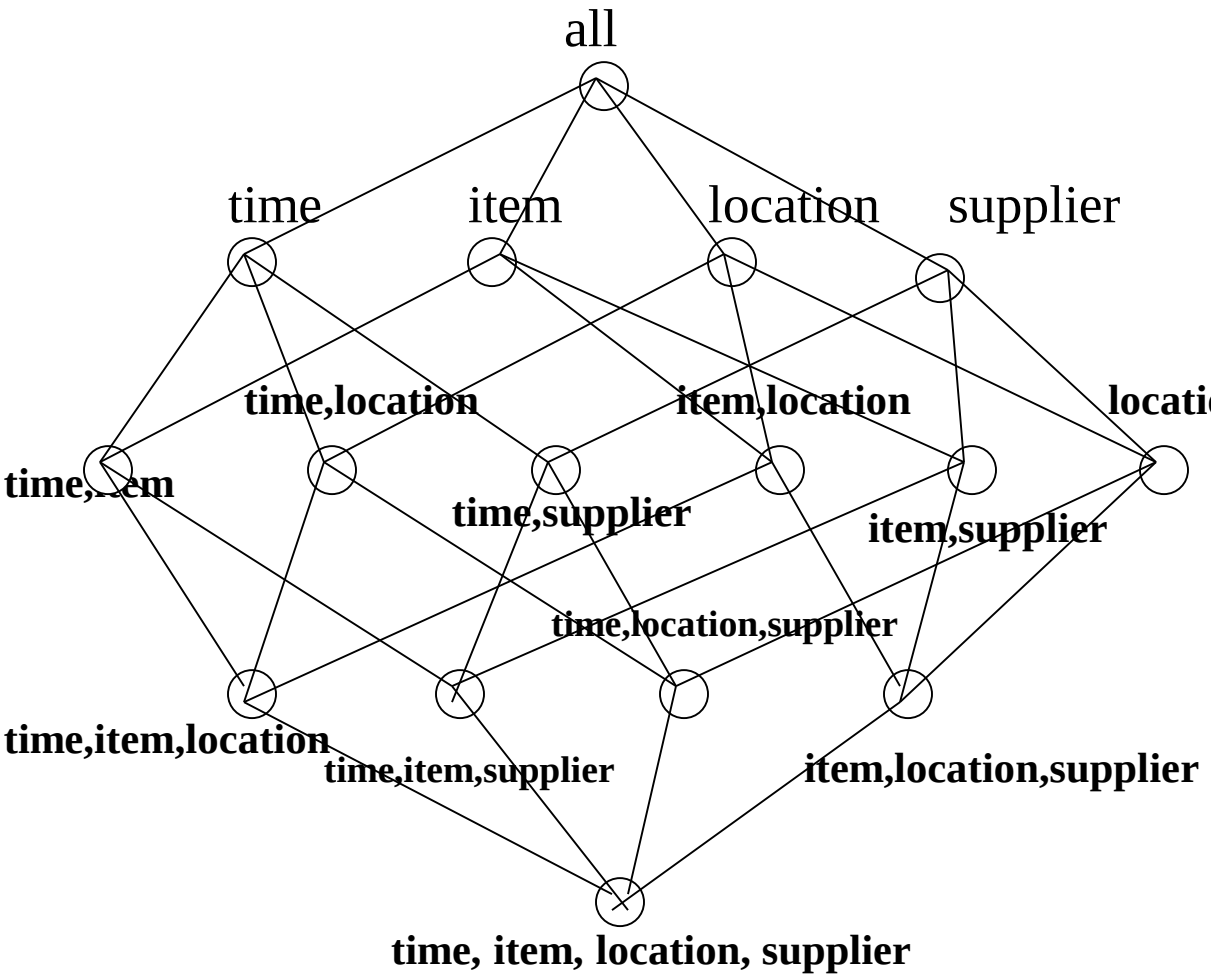
- در سطح پایین، ما فقط یک سری measure‌ها را مشاهده است. ← یک سری مقادیر کلی بدون درگیر کردن dimension‌ها.

مثال بعدی ایتم: می خوایم ببینیم کدوم ایتم بیشتر فروش رفته مثلا بگو دوغ الیس چقدر فروختی یا نوشابه چقدر فروختی که این 2 تا جایگاه مختلف است ک این میشه اون بعده ینی اون مقدار کل فروش رو بیا بشکون به دسته بندی سطح بالای ایتم ها --> الان اینجا ایتم dimension است ولی مثلا نوشابه یا دوغ ینی می خوایم از دو منظر متفاوت اینو ببینیم
معمولا dimension ها یک سلسله مراتبی در اختیار ما قرار میدن مثلا میگیم کل فروش چقدر بوده که اینجا اصلا dimension درگیر نیست مثلا کل فروش 2 میلیارد تومن است و حالا میخوایم از جنبه لوکیشن نگاه بکنیم که این لوکیشن یک dimension است حالا یک لایه میایم پایین تر و میگیم از این 2 میلیارد تومن چقدر توی هر استان فروختی ممکنه فقط توی 10 تا استان شعبه داشته باشه پس فقط این 10 تا رو نشون میده مثلا یکی از اون 10 تا تهران است حالا بازم میایم لایه پایین تر مثلا می ریم سراغ استان تهران و میگیم توی شهرهای مختلفش چقدر فروخته شده و به همین صورت ادامه داره که اینا میشه سلسله مراتب --> اینا همه از منظر لوکیشن است
نکته: ایتم یک dimension توی بحث فروشگاه می تونه باشه مثلا میخوایم مقدار فروش رو از منظر ایتم بررسی کنیم
خلاصه انبار داده و cube هم همین است : ما میخوایم از ابعاد مختلف یک ابزاری در اختیار مدیر قرار بدیم و بعد از این ابعاد خلاصه و چکیده می خواد شاخص هایی رو بررسی کنه پس این استفاده خیلی سطح بالا است
برای اینکه یک مژری رو بررسی کنیم می تونیم dimensionهای مختلفی درگیر بکنیم یا یک dimension درگیر بکنیم یا اصلا چیزی رو درگیر نکنیم مثلا اونجایی که dimension درگیری نشده این می تونه باشه که مثلا مقدار کل مبلغ فروش یا تعداد کل تراکنش ها یا مقدار سود یا.. اینا میشن چیزایی از جنس مژر و اینجا dimension درگیر نشده

نکته: وقتی میگیم تعداد کل تراکنش ها رو بهمون بده بدون این که dimension درگیر بشه با وقتی که میگیم مقدار مانده کل حساب ها رو بهمون بده یک تفاوت ریزی بینشون هست --> اینجا یک بحثی پیش میاد به اسم last non empty --> برای اون تعداد کل تراکنش ها می تونه بحث last non empty مطرح بشه یا می تونه مطرح نشه به چه صورت؟ وقتی میگیم تعداد کل تراکنش ها ینی میتونه توی کل عمر این بازار که ممکنه 10 سال بوده باشه کلا بگه چند تراکنش انجام شده و اینجا هیچ dimension درگیر نکردیم پس اینجا باید کل هیستوری رو نگاه کنه که چند تراکنش انجام شده ینی در واقع باید sum بزنه ولی وقتی که میگیم مانده کل حساب های بانک رو بگو این last non empty ینی براساس هر سپرده ای یک مانده ای داریم و این مانده هم دیروز یه چیزی بوده و ما نمیخوایم با مانده امروز رو با مانده دیروز جمع بکنی فقط میخوایم اخرین مقدارش رو در نظر بگیری

Cube: A Lattice of Cuboids

اینا اسم سطح است:



0-D(apex) cuboid

1-D cuboids

2-D cuboids

3-D cuboids

4-D(base) cuboid

ما یک موقع میخوایم یک مژری رو مثل همین مبلغ کل فروش رو نگاه بکنیم و اگر هیچ dimension رو درگیر نکرده باشیم به این میگن apex پس اینجا وقتی میگیم کلش ینی جمع رو میخوایم یا میانگین رو میخوایم یا ... --> اینجا ینی توی all ما group by نداریم ولی aggregation داریم --> این تهشو از یکی بپرس حتما؟؟؟

D-1: یک موقع میگیم فقط تایم رو بده یا فقط ایتم رو بده یا ... --> اینجا یک dimension داریم

نکته: از منظر زمان بیا بگو توی سالهای مختلف چقدر فروختیم این میشه group by روی time ینی group by روی یک dimension داریم

توی سطح دوم ینی D-2 ما دوتا dimension داریم --> پس دو نوع dimension مختلف داره group by میشه مثلا به ما بگو در سال 1391 چخبر بوده و اینو روی نوشیدنی بگو

توی سطح 3: 3 تا group by داریم

توی سطح 4 یا base روی همش group by داریم

نکته: اطلاعات مژرها رو توی فکت تیبل ذخیره میکنیم --> مثلا مبلغ فروش یک مژر است و این مژر توی جدولی هست از جنس فکت

all: مبلغ فروش رو توی یک جدول ذخیره کردیم به اسم فکت و وقتی که میخوایم این جدوله رو براساس زمان مثلا group by بکنیم یا براساس ایتم یا ... توی این جدوله برای این کار باید چی داشته باشیم؟ اخرین سطح رو باید داشته باشیم --> توی سطح پایه همه چیز رو داریم ولی داریم group by میکنیم و توی سطوح مختلفش می شکونیم براساس این که چی داریم ؟؟

توی رکوردهای فکت باید کلیدهای time , item , location , supplier باشه و همینطور مبلغ هم باید باشه --> این میشه فکت

توی فکت کلید اون dimension رو می ذاریم و این کلید باید ریزترین اون باشه ینی مثلا برای time این کلید نمیتونه سال باشه پس کلید dimension میشه روز

Conceptual Modeling of Data Warehouses

- Modeling data warehouses: dimensions & measures
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - Fact constellations: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

-

مدلینگ روی بحث انبار داده:
مدل استار: یک فکت تیبلی وسط است و dimension ها با کلیدهاش وصله بهش
نکته: یک فکت نداریم فقط --> پس هر فکتی که میخوایم یک جدول می ذاریم وسط و مژرهایی که میخواد داخلش هست و dimensionها مستقیم بهش وصلن (منظور از وصل بودن این است که کلیدهای dimensionها داخل جدول فکت هست)
نکته: طراحی هایی که توی پروژه انجام میدیم باید از این مدل باشه و کاری به بقیه مدل های دیگه نداریم
پس دیفالت ما توی این درس مدل استار است مگر اینکه دقیقا از ما مدل Snowflake را خواسته باشد
مدل Snowflake: خیلی با این مدل کاری نداریم - خلاصه مدل استار این میشد که اگر یک فکت کشیدیم هر dimension مستقیم با فکت ما در ارتباطه از طریق کلیدش ولی توی مدل Snowflake میگه ممکنه dimension رو دولایه ببینیم ینی این dimension به فکت وصل است ولی خود این dimension به یه چیز دیگه هم وصله
مدل galaxy: فرض میکنیم مدل استار است و ممکنه توی انبار داده ما 50 تا فکت داشته باشیم و این 50 تا فکت به dimension هایی متصل است پس به این مجموعه انبار داده میگیم galaxy
پس در کل یک مدل خاصی نیست

Dimensional Modeling

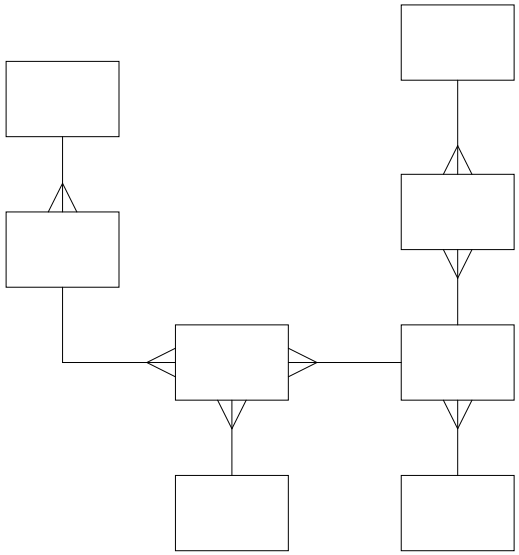
- Dimensional modeling = data warehouse modeling technique
- 2 types of tables: facts and dimensions.
- A fact table contains one or more measures (usually numerical) of a subject that is being modeled for analysis.
- Dimension tables contain various descriptive attributes (usually textual) that are related to the subject depicted by the fact table.
- The intent of the dimensional model is to represent relevant questions whose answers enable appropriate decision making in a specific business area

-

Dimensional Modeling همون DW modeling است
جدول ها توی طراحی استار دو نوع هستن:
فکت ها و dimensionها
توی فکت ها مژرها + کلید dimensionهایی که میخوایم این مژرها رو از اون منظرها ببینیم
شامل یک یا چندتا مژر هستن که این مژرها معمولا از جنس numerical یا عددی هستن و
dimensionهایی که میخوایم ببینیم رو عملا وصل میکنن
نکته: مژرها KPIهامون رو میسازن
KPI معمولا نسبت چندتا مژر است
dimensionها میشه اون جدول های پایه ای که می خوایم مژرها رو از اون جنبه ها بررسی کنیم
مثل تایم --> اطلاعات پایه رو داخل dimension میذاریم
مژر از جنس عددی هست بیشتر چرا؟ یک بحثش میشه aggregation ها و
یک بحث دیگش هم بیزینسی میشه --> 1- چون یکسری شاخص رو می خوایم و 2- وقتی که
میخوایم این شاخص رو نگاه بکنیم براش aggregation می زنیم و aggregation معمولا روی
یک چیز عددی است مثلا شماره تلفن می تونه یک مژر باشه؟ نه چون اصلا بدردمون نمی خوره
مثلا جمع شماره تلفن یا چی مثلا کلا این کارایی برامون نداره پس نمیشه براش زد کلا شاخصی
روی شماره تلفن نمی خوایم داشته باشیم

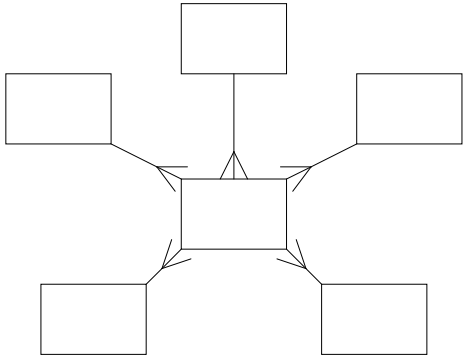
مدل ذخیره داده

Operational System



ER Diagram

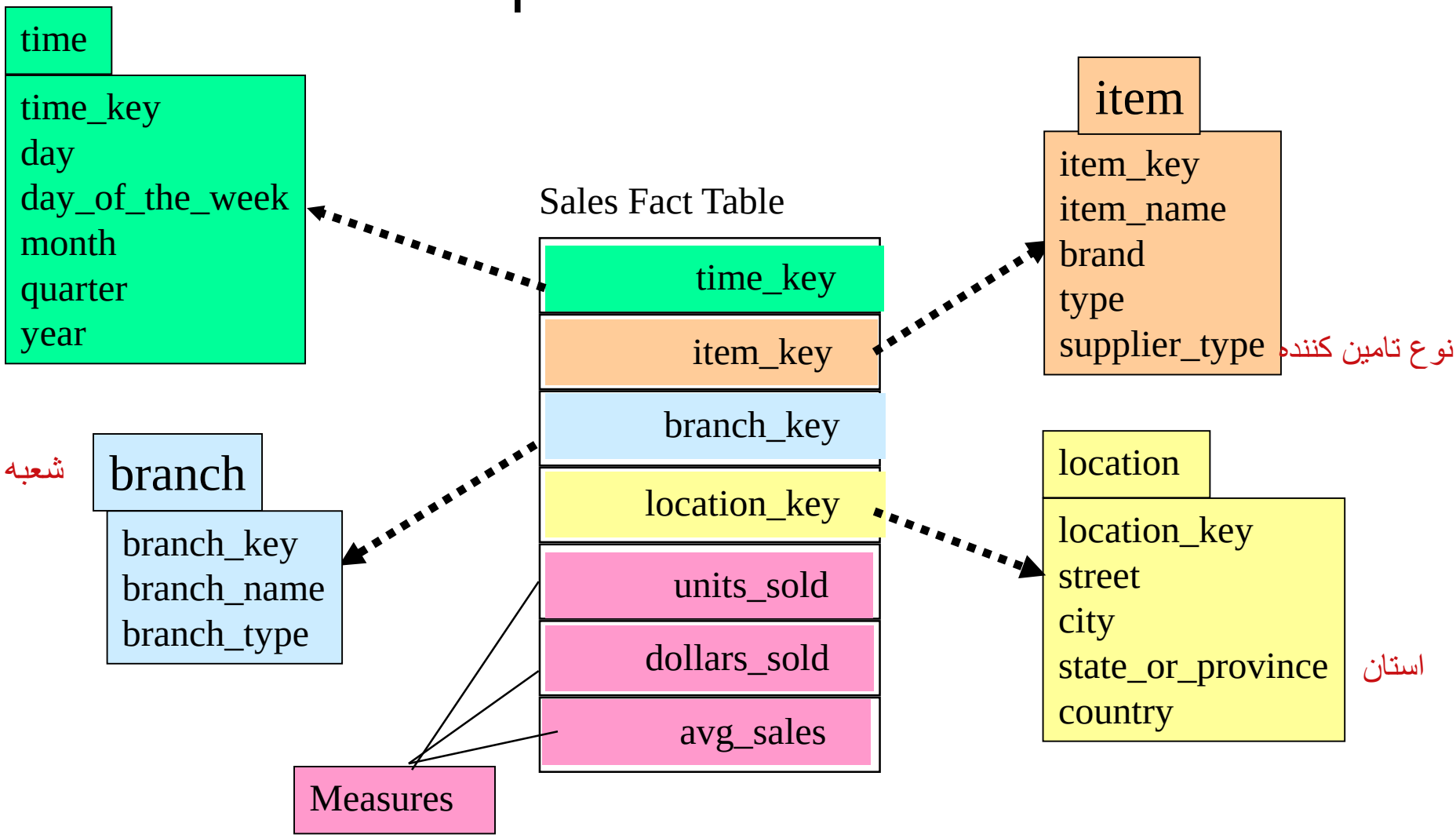
Data Warehouse



Star Schema

توی مدل ER Diagram میگه این جدول با اون جدول ارتباط داره و اون یکی مثلا با اون یکی و..
ولی توی مدل استار به این صورت نیست یک جدول وسطه که بهش میگیم فکت و همه با این فکته در ارتباط هستن

Example of Star Schema

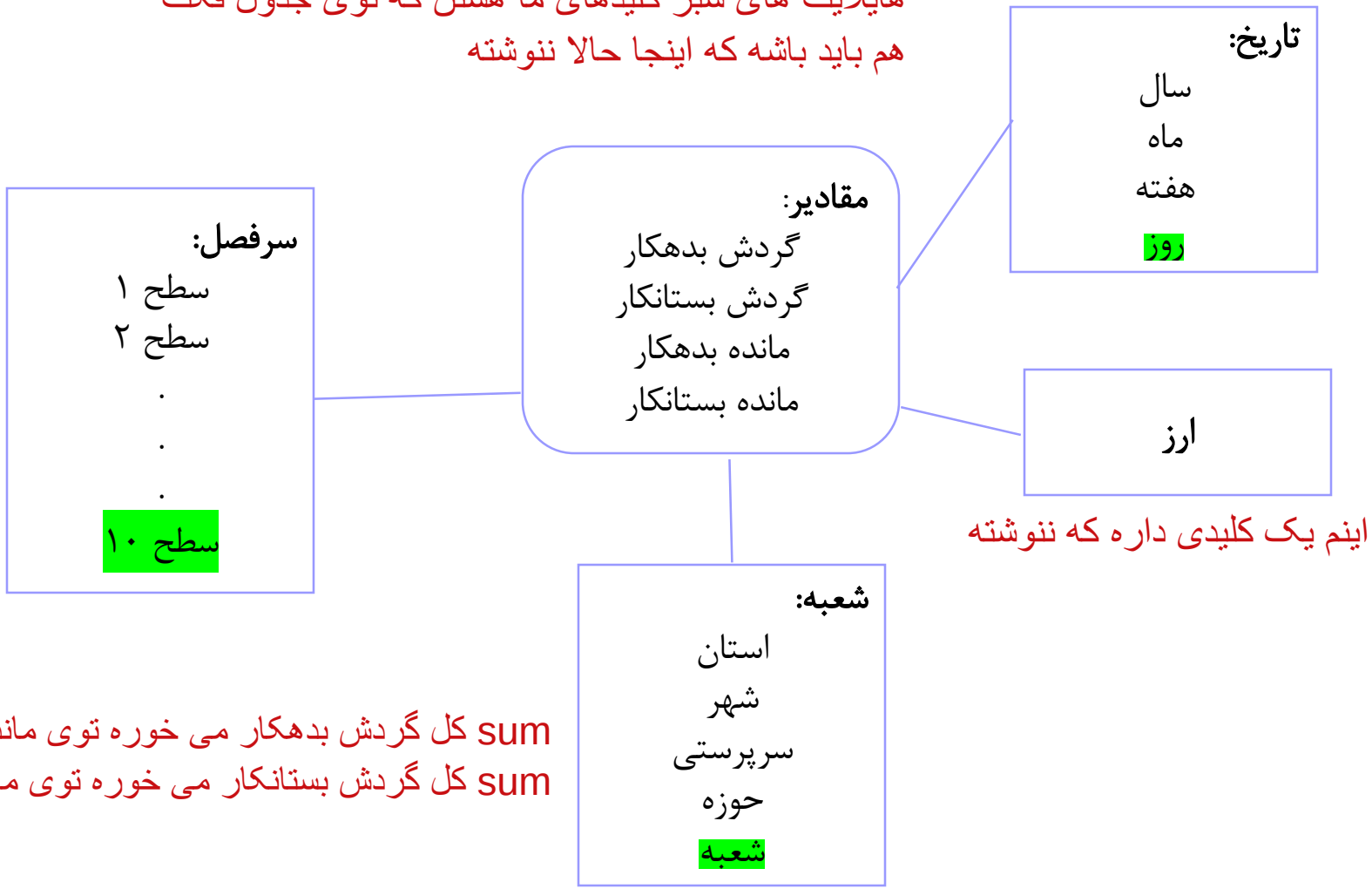


مثال کامل از مدل استار:
از جنس فکت فروشگاه است
اینجا 4 تا dimension داریم
نکته: وقتی میگیم location_key ینی ریزترین بخش این قسمت است و این کلیده هم یکتا است و برای بقیه dimension های دیگه هم به همین صورته که بالا گفتیم
time_key مثلا میشه خود کد محصول اگه خیلی ریز شده باشیم
نکته: این که بگیم برای فکت کلید داریم خیلی توی انبار داده معنایی نداره
اینجا 3 تا مژر داریم و می خواد این 3 تا مژر رو از منظرهای مختلف ببینه مثلا میخواد ببینه
units_sold چقدر بوده --> اگر بخوایم اینجا apex ببینیم میشه:
select sum(units_sold)from Sales Fact Table
و اگر بخوایم base رو ببینیم میشه:
select * from Sales Fact Table
مثال برای این مدل استار:
یک رکورد از ایتم:
1 - دوغ - الیس - لبنیات - لبنیاتی
یک رکورد از لوکیشن:
1 - امام خمینی - شهر اصفهان - اصفهان - ایران
یک رکورد از برنچ:
1 - شعبه 1 - فیزیکی
یک رکورد از تایم:
1402/1/1 - 1 فروردین - شنبه - فروردین - بهار - 1402
فکت میشه:
از بالا به پایین فکت میشه : راست به چپی که الان می نویسم:
1402/1/1 1 1 1 100(مثلا 100 واحد فروخته شده) 10000 100

مثالی از apex:
sum تعداد فروش
مثالی از base:
100 10000 100 --> برای این dimension دخیل نمیشه

مثال مدل STAR

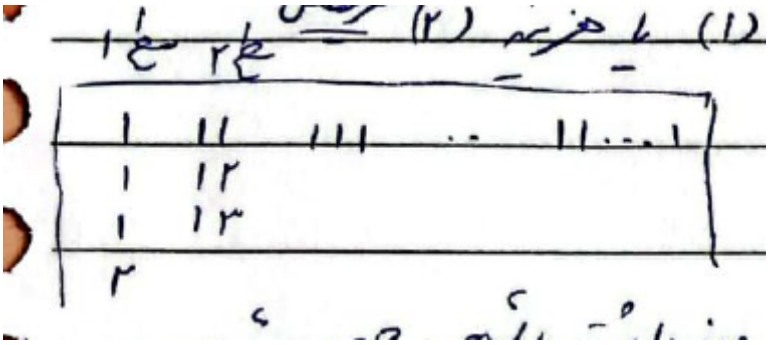
هایلایت های سبز کلیدهای ما هستن که توی جدول فکت
هم باید باشه که اینجا حالا ننوشته



اینم یک کلیدی داره که ننوشته

sum کل گردش بدهکار می خوره توی مانده بدهکار و
sum کل گردش بستانکار می خوره توی مانده بستانکار

مثال:
حسابداری --> برای درامدها و هزینه ها یک سری سلسله مراتب (همین جدول سرفصل است) ایجاد میکنن:
انواع مختلفی از درامد و هزینه داریم
یک همچین جدولی داریم: اگر توی تراکنش مثلا خورد 11 ما می فهمیم از کدوم نوع درامد است --> این جدول پایین صفحه رو ما داریم --> این مفهوم دفتر کل سمت دیتابیس operational
برای هر کدوم از این سرفصل ها توی ادبیات حسابداری گردش بدهکار و بستانکار معنا داره
یک سرفصل معمولا ماهیتش یا بستانکاره یا بدهکار
وقتی یک سرفصلی داره گردش می خوره مانده هم می تونه براش معنا داشته باشه
تا این حد محاسبات مهمه چون توی امتحان میاره --> صفحه بعدی با مثال گفته



مثلا توی این جدوله توی سطح اول 1 نشان دهنده درامده و 2 نشان دهنده هزینه بعد توی سطح بعدی 1 و 2 اولیش همینو نشون میده و عددای بعد از این مثلا می تونه نشان دهنده نوعش باشه و برای بقیش هم به همین صورته واسه همین همه اطلاعات رو توی سطح 10 داریم

مثال سوپر مارکت:
سوپر مارکت یکسری درامد و یکسری هزینه داره
درامدش مثل همین فروختن پاکت شیر میشه --> این یک نوع درامد است
این شیر داره می فروشه یا نوشابه می فروشه یا... این اقلام ریز که داریم در نظر می گیریم سطح اخر سرفصل میشه ینی سطح 10
اگر شیر داره میفروشه یا نوشابه می فروشه اینا توی سطح بالاتر یک اشتراکی باهم دارند و.. و تهش می شه درامد
مثلا لبنیات سطح 5 است و سطح 6 میشه شیر و دوغ و..

برای هر کدوم از این سطوح --> برای خرید اقلام داریم هزینه می کنیم ینی داریم گردش بدهکار میزنیم و برای فروش اقلام میشه بستانکار --> اینجا داریم در مورد ریزترین بخش حرف میزنیم چون تراکنشی که ثبت میشه به سمت ریز ریز می خوره

مثال: برای هر روز داره گردش بستانکار هر سر فصلی رو نگه میداره و اگر توی تراکنش ها 10 تا تراکنش بستانکار برای سرفصل سطح 10 خورده باشه توی فکت یک رکورد داریم چون
جمعشون میزنه و توی فکت به عنوان گردش بستانکار می زنه - گردش بدهکار چون نداشتیم میشه صفر - مانده بدهکار میشه همون مقدار قبلی که داشت - مانده بستانکار هم میشه گردش بستانکار امروز + گردش بستانکار روزهای قبل

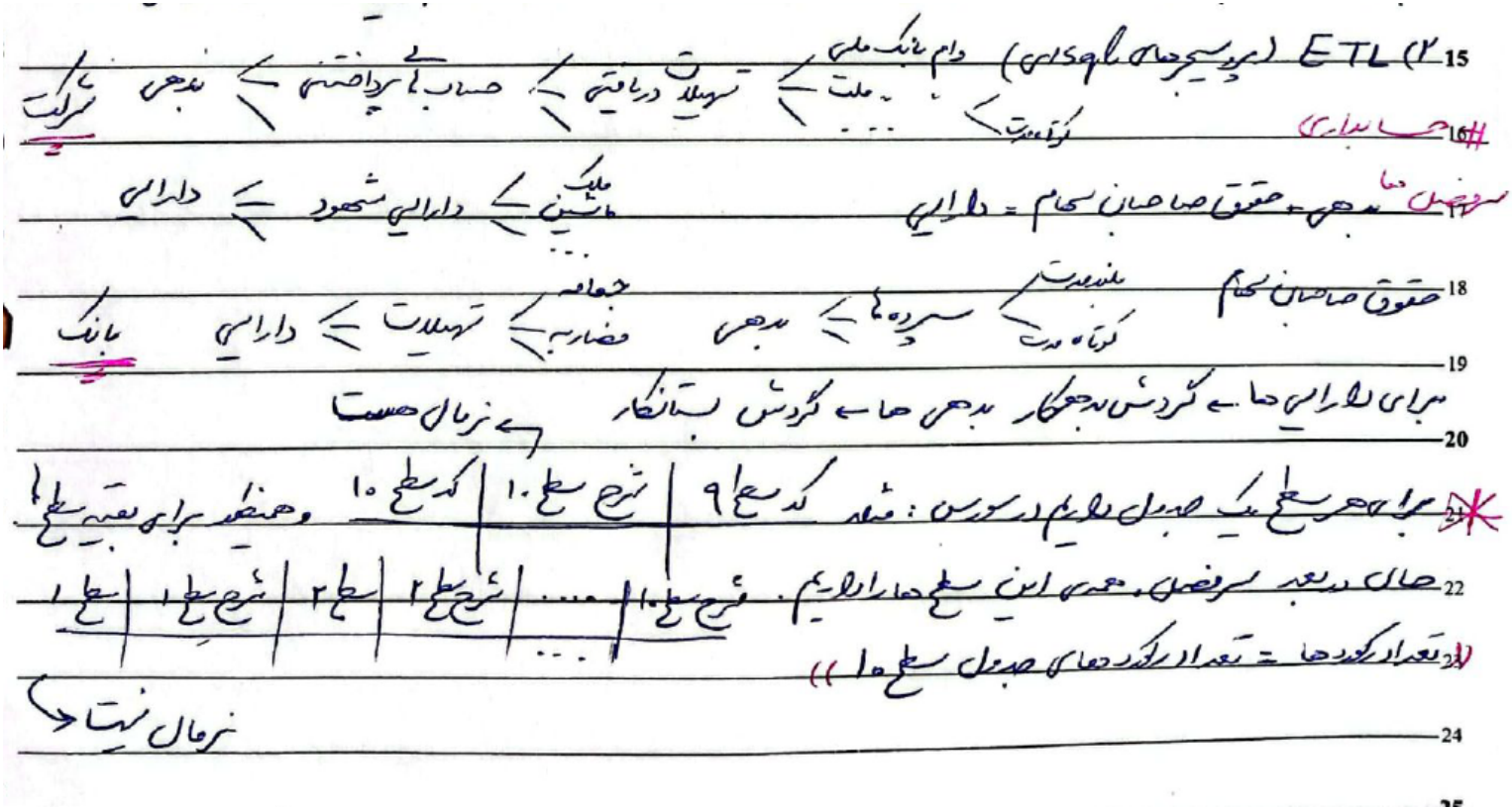
نکته: چرا مژرها عددی هستن؟ از لحاظ بیزینسی ببین: چون می خوایم نیاز شاخص های عملکرد کلیدی سازمان رو جواب بدیم --> KPI به صورت عرف نسبت چندتا عدد است
KPI که میخواد بررسی بشه توی سازمان اینا ماهیتا عددن ینی نسبت یکسری عدده که داره بررسی میشه توی زمان --> شاخص ما عددی هست و مژرها کمک میکنه که ما رو به اون شاخص برسونه پس عددی میشه در عین حال چون اینجا می خوایم agregation اینجا انجام بدیم و برای agregation چیزی غیر از عدد معنی نداره

یک مثال دیگه: --> شکل صفحه بعد...
جزئیاتش رو لازم نیست بدونیم
یکسری اصول اولیه حسابداری وجود داره: (در همین حد بلد باشیم کافیه)
دارایی هایی یک سازمان = حقوق صاحبان سهام + بدهی--> این یک اصل اولیه پایه حسابداری است --> ینی چی این؟ فرض کن دو و سه نفر رفتیم شرکت زدیم و یک میلیارد پول اوردیم و رفتیم یک ملک خریدیم برای فضای شرکت برای این کار رفتیم وام هم گرفتیم پس یک میلیارد پول داشتیم و دو میلیارد هم وام گرفتیم --> پس 3 میلیارد تومن داریم و رفتیم یه جایی رو اجاره کردیم
دارایی ما اینجا اون ملکه است که ارزشش 3 میلیارده و خودمون 1 میلیارد اوردیم و دو میلیارد هم بدهکاریم --> این میشه فرمول بالا الان --> این اساس حسابداری است
حالا توی سطوح مختلف حسابداری این مفاهیم رو دارن کنترل می کنن و گزارش می گیرن و از لحاظ مالی می خوان بسنجن و agregation هایی انجام بدن توی بحث حسابداری --> هرجایی که بحث مالی داریم این کارا انجام میشه
قبلا گفتیم توی 10 تا سطح می تونیم ببینیم این مفاهیم رو حالا توی سطح اولش همین 3 تا است ینی میشه دارایی و بدهی ویک حقوق صاحبان سهام که جمع اینا باید با هم بخونه
حالا توی دارایی ریز میشیم مثلا ینی توی لایه بعد ما دارایی های مشهود و نامشهود داریم
حالا توی این دارایی هایی مشهود می تونه زیرمجموعه هایی باشه مثل ماشین، ملک
و همینطوررری می ره جلوتر و هی می تونه بشکنه
بدهی خودش انواع مختلفی میشه و می ره جلو و یکی از بدهی ها حساب های پرداختنی است و حساب های پرداختنی یکی از انواع تسهیلاتی است که دریافت کردیم --> اینایی که باید پرداخت کنیم جز این بدهی ها است
اینایی که گفت میشه سطوح مختلف

مثال بعدی: شکل صفحه بعدی...
روی بانک است:
توی مثال قبل گفتیم تسهیلات می گیره و بعد باید پولشو بده ینی این بدهی است ولی بانک وقتی وام میده باید پول بگیره پس تسهیلات بانک جز دارایی هاش است ینی مال بانک است
سپرده هایی که از مردم میگیره بدهی های بانک میشه چون باید بعد به مردم برگردونه

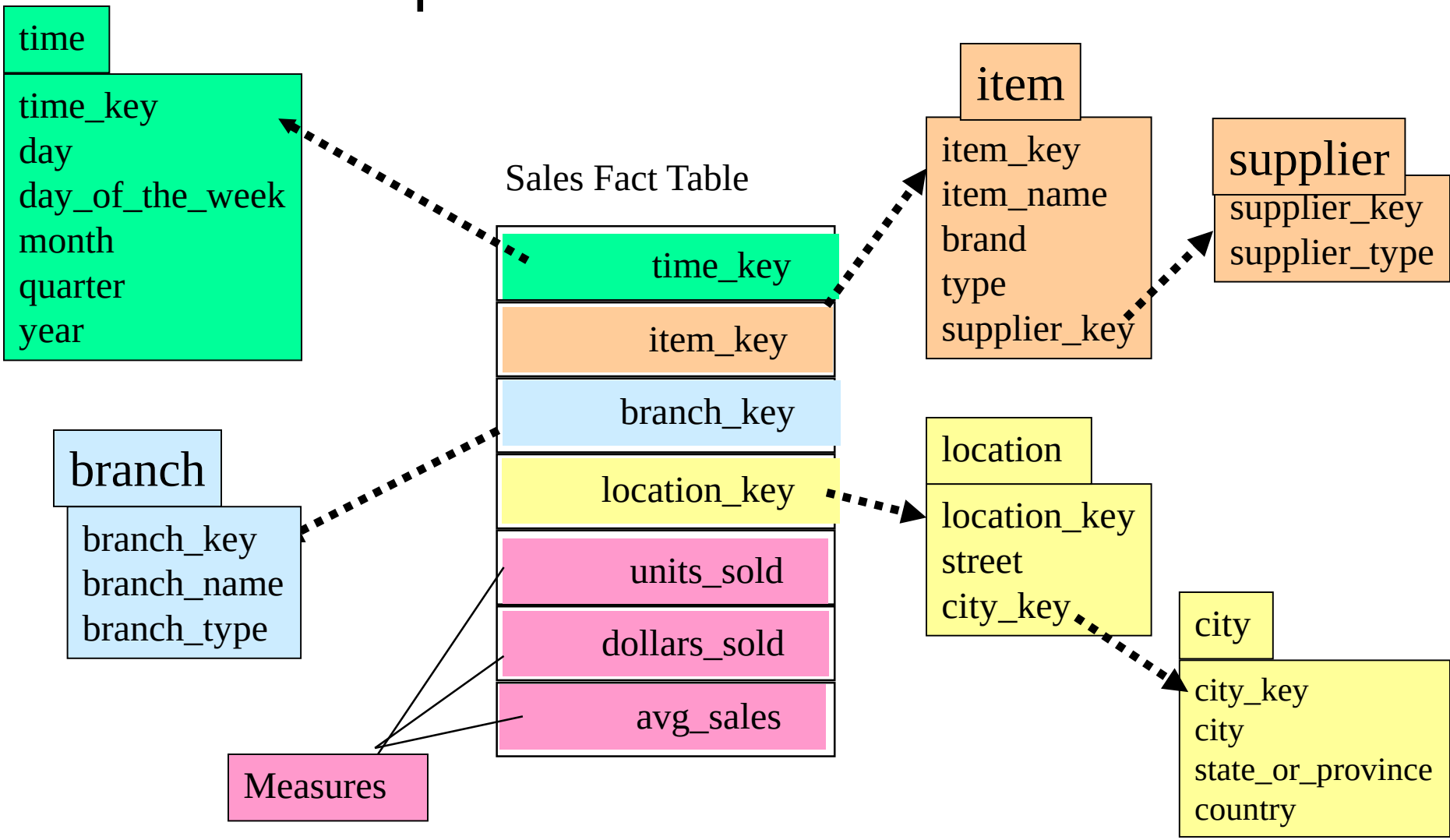
حالا برای همین مثال اسلاید که مثال مدل STAR بود میشه:
برای سرفصل های حسابداری:
سطح 1 میشه --> دارایی و بدهی و حقوق صاحبان سهام میشه
نکته: اگر سطح 10 رو داشتیم می تونیم تا سطح 1 رو به دست بیاریم
نکته: توی بحث حسابداری یک اساسی است که دارایی ها ماهیتا بدهکار هستن پس برای دارایی ها ما گردش بدهکار می زنیم پس براش مانده بدهکار هم مقدار میگیره و برای بدهی ها، گردش بستانکار می زنیم پس براش مانده بستانکار مقدار میگیره (اینو قبول کن توی همه جا)

نکته: اگر بخوایم توی بحث حسابداری طراحی با ادبیات دیبی 1 داشته باشیم --> اولش باید یک سطح ریز ریز داشته باشیم ینی سطح 10 که این سطح یک کد داره ویک شرح داره --> پس توی سطح 10 می تونیم یک جدول داشته باشیم که سه تا ستون داره یکیش کد میشه و یکیش شرح میشه
و یکیش هم سطح 9 میشه
یک جدول دیگه ای داشته باشیم که این سطح 9 رو با کد و شرحش نوشته باشه و یک کلید خارجی هم داریم که مال اینه که بگه کد سطح 8 چیه ؟؟
و برای بقیه سطح ها هم همینطور
سوال: الان سر فصل ها باید به 10 تا جدول وصل بشه؟ نه --> این یک جدول است و اینو توی ETL ها هندلش کردیم و جوین زدیم روش تا برسیم به این جدول



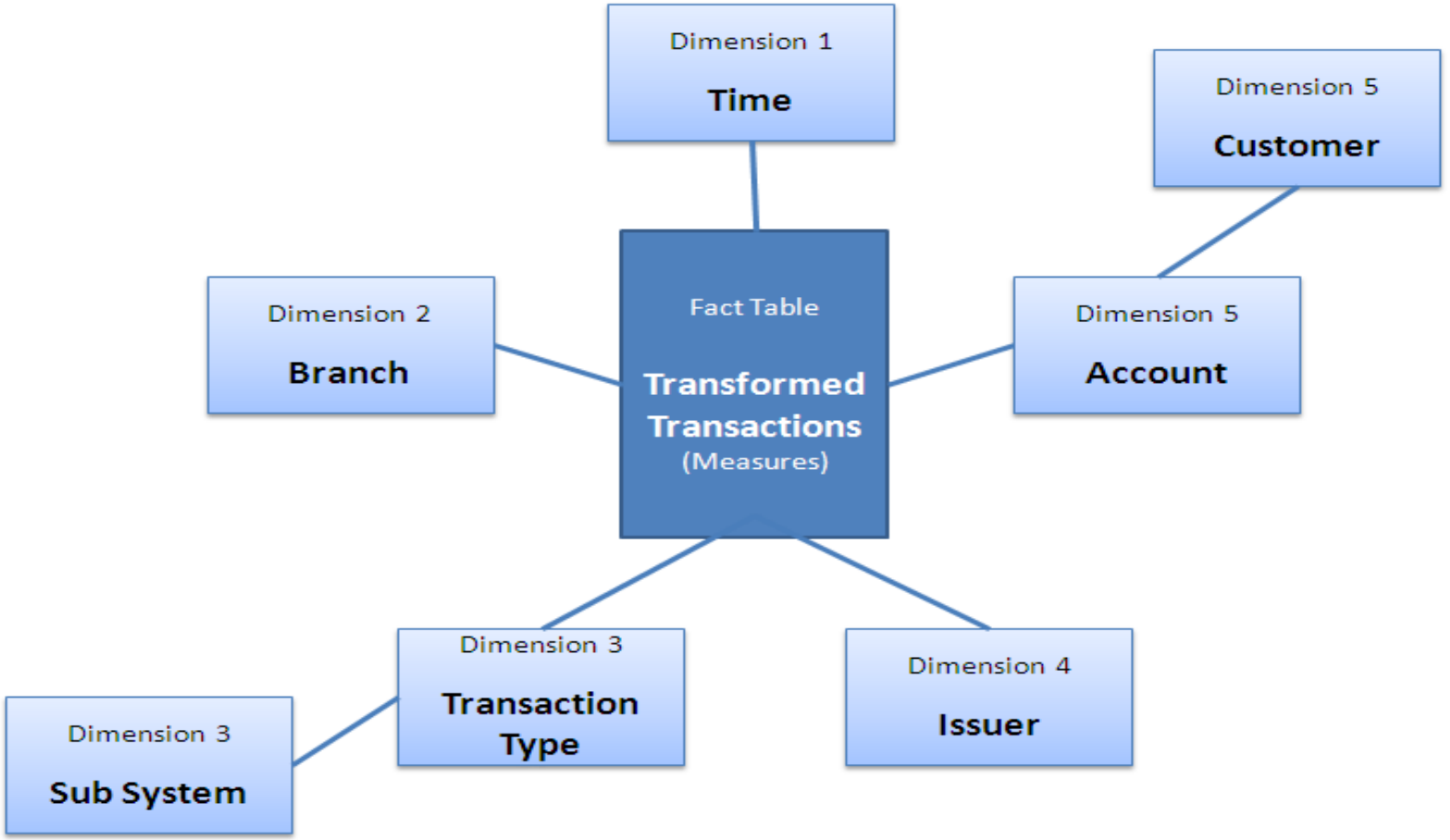
برای یک فروشنده مغازه:
بستانکار--> مغازه دار پول بگیره
بدهکار --> مغازه دار پول بده

Example of Snowflake Schema



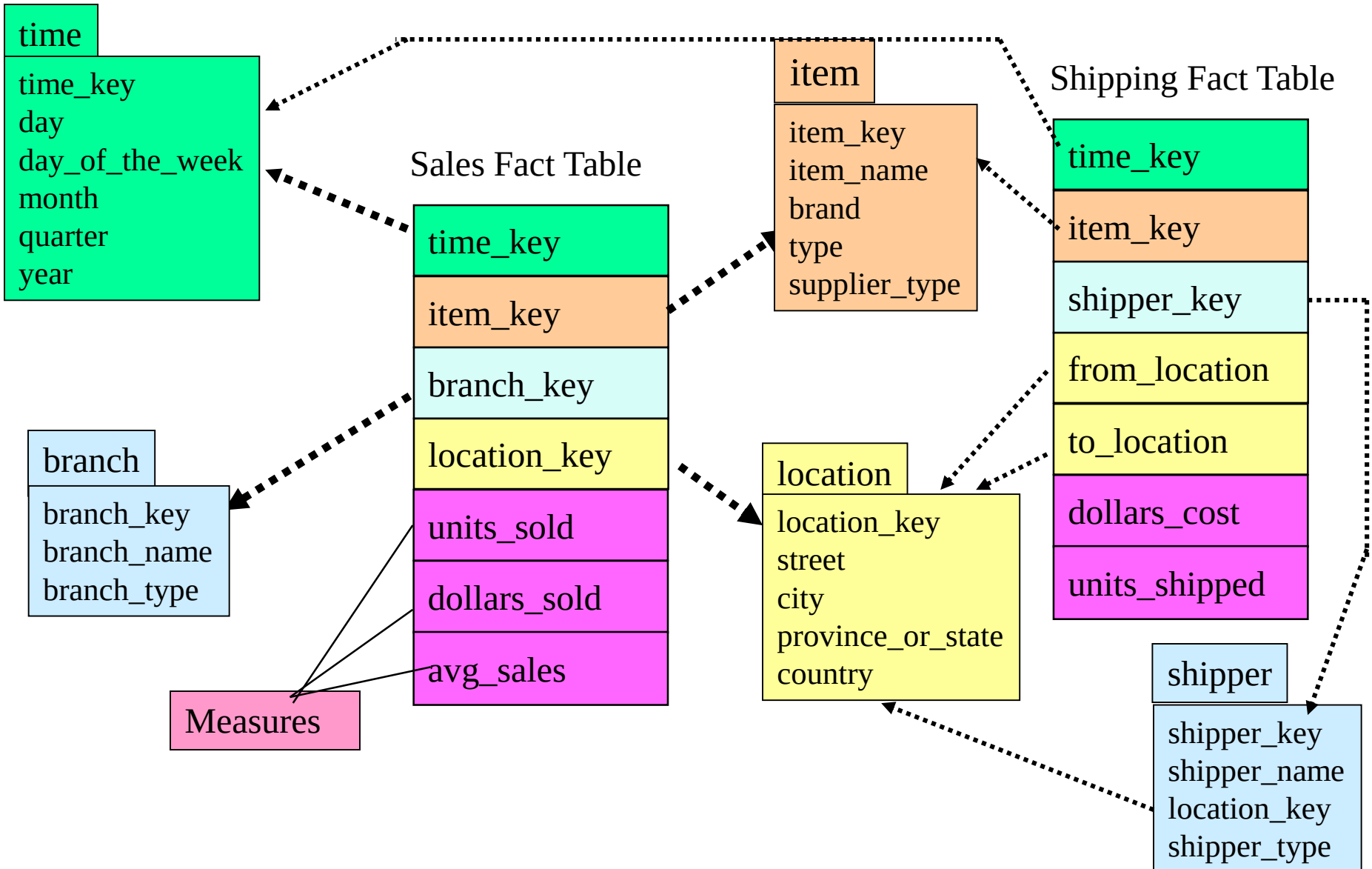
مدل Snowflake:
ما می تونیم گاها یک dimension رو به جای اینکه همشون رو توی یک سطح بدیم دوتا سطحش می تونیم بکنیم
چرا این مدل جالبی نیست؟ چون اینجا جوین اضافه میکنه بخاطر همین مدل خیلی کاربردی نیست و در عمل یکسری پیچیدگی هم اضافه میکنه بهمون

Snowflake Schema مثال



مثال: یک جدول فکت در مورد سپرده است:
ما میخوایم بدونیم این تراکنشی که اتفاق افتاده سپرده مال کدوم مشتری است --> این که این سپرده مال کدوم شخص و اطلاعات اون شخص چیه رو داخل یک جدول دیگه اورده
اگر می خواستیم این مدل رو استارش کنیم باید همه فیلدهای customer رو بیاریم توی account و کلیدش تغییری نمیکنه ینی همه کاستومر میاد توی اکانت اره؟؟؟ حتی کلیدش؟ --> توی این حالت تکرار دادمون ممکنه یکم بیشتر بشه چون یک کاستومر ممکنه چندتا اکانت داشته باشه

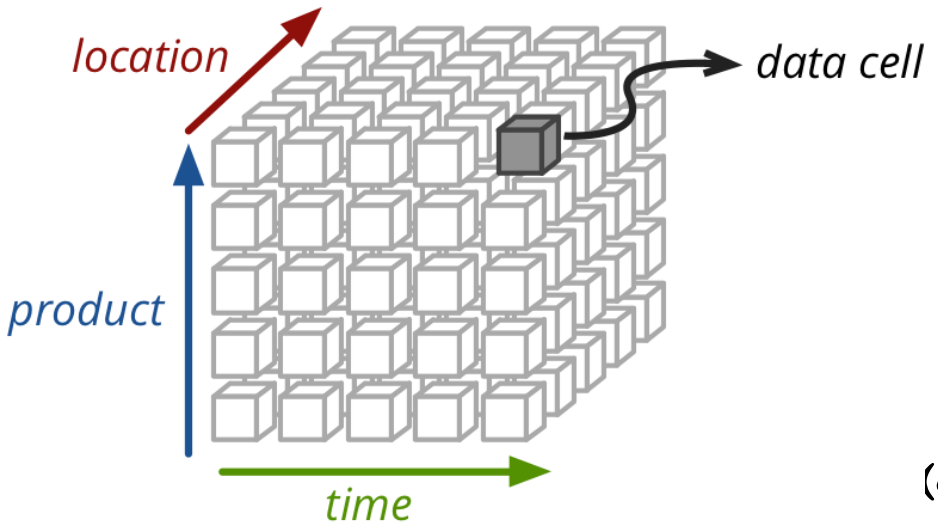
Example of Fact Constellation



به این فضا میگن فضای galaxy
ایا می تونیم از این dimension مشترک از Sales Fact برسیم به Shipping Fact یا برعکس؟ نه نمی تونیم
نکته: جوین بین فکت ها توی انبار داده معنایی نداره

نحوه نگهداری داده ها در کیوب

- مدل داده ای چند بعدی **(Multi Dimensional)**



- **Cube** (مکعب): محل قرار گرفتن داده

- جداول **Fact** (واقعیت)
- **Measure** (مقادیر)

- جداول **Dimension** (بعد)
- **Hierarchy** (سلسله مراتب)
- **Drill Down**

سلسله مراتب توی dimension معنا پیدا میکنه و همینطور بحثی مثل Drill Down
توی نگاه چند بعدی داخل این cell که توی تصویر نشون داده شده یک مقداری درونش ذخیره شده
مثلا مقدار فروش در فلان شعبه و از نوع فلان و... ==> این داره یک مقداری به ما میده مثلا
مقدار فروش رو به ما میده که میشه اطلاعاتی ک از جنس مژر است
حالا این مقدار ذخیره شده رو می تونیم از این سه تا جنبه بهش دسترسی پیدا بکنیم:
مثلا برای این مقدار داخل فکت چه dimensionهایی و مژرهایی داریم؟ برای dimensions:
time , product , location میشه + چندتا مژر
بحث سلسله مراتب: کلید تایم توی فکت هم می خوره ولی می تونیم روی بعد زمان اطلاعاتمون رو
aggregate بکنیم مثلا گفتیم به صورت روز داخل فکت اینو بریز حالا میتونیم به ماه ببینیم و به
سال ببینیم و ...

OLAP (CUBE)

- ✓ **OLAP (On-Line Analytical Processing)** با استفاده از تجمیع مقیاس‌ها و شکستن آنها بر روی ابعاد مختلف کسب‌وکار، یک شمای متناسب با نیازهای گوناگون بخش‌های مختلف در اختیار بانک قرار می‌دهد.

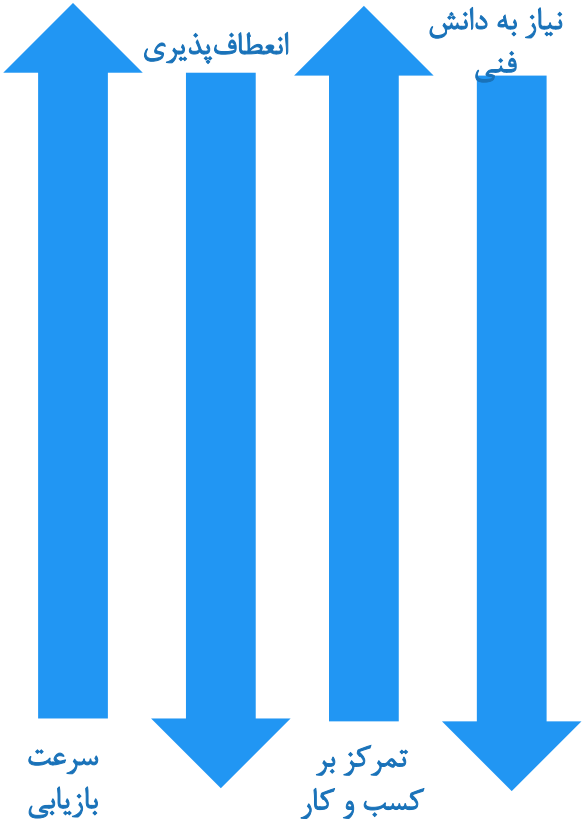
- ✓ **Cube** قالب نگهداری داده ها در مدل چند بعدی است.

- ✓ با استفاده از **Cube** داده ها به آسانی و به سرعت می تواند در اختیار کاربران غیر فنی قرار گیرد.

olap میشه ابزاری که به ما این امکان رو میده که این ریز شدن یا aggregate شدن را بتونیم راحت استفاده بکنیم ازش --> ینی مژرهایی که داخل یک کسب و کار مهم هستن رو با استفاده از dimensionهایی که براش تعریف کردیم بتونیم مشاهده بکنیم

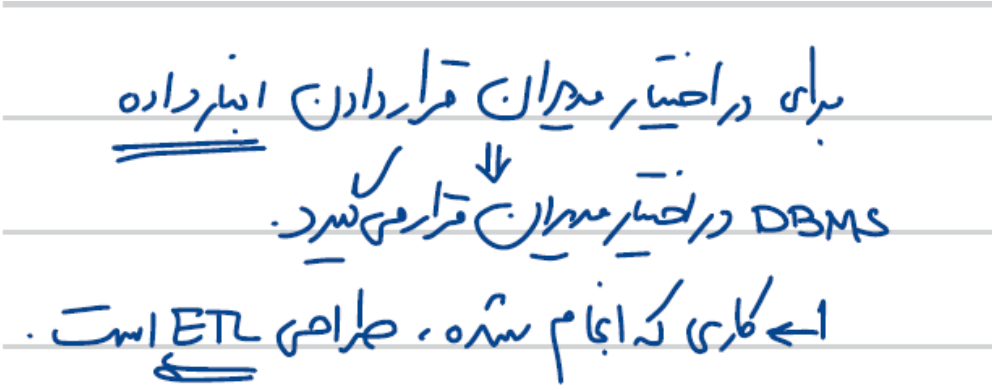
اگر cube رو پیاده سازی کرده باشیم و داشته باشیمش مزیتی که داره اینه که کاربری که کاربر فنی نیست خیلی راحت می تونه از اون cube استفاده بکنه و دیتایی که مدنظرش هست رو مشاهده بکنه

لایه‌های هوش تجاری



مقایسه بین لایه ها:
یک لایه DW داریم، یک لایه olap و یک لایه reporting و یک لایه applications
توی olap مون cube قرار داره --> وقتی انبار داده را ایجاد میکنیم و جدول ها رو می سازیم با ادبیات فکت و dimension میتونیم بیایم dimension و فکت ها رو توی cube بذاریم

هرچی می ریم به سمت بالاتر سرعت بازیابی اطلاعات معمولا بیشتر میشه و از طرفی انعطاف پذیریمون کم میشه --> توی انبار داده همه اطلاعات ریز رو داریم ینی ریز همه اطلاعات توی اون فکت و dimension ها هستش و ممکنه توی این olap که میریم همه اون فکت و dimension ها رو اصلا نیاورده باشیمش و بازم هرچی می ریم بالاتر از این ریز بودن اطلاعات ممکنه که دور بشیم --> توی DW انعطاف از لحاظ دستری به داده بیشتر است
و چون aggregationهامون بیشتر است سرعت بازیابی می ره بالا
اگر یک گزارش سطح بالا بخوایم ینی سطحی که aggregation داخلش بیشتر است بهتره از لایه olap استفاده بکنیم ولی اگر بخوایم جزئیات رو مشاهده بکنیم از لایه DW استفاده میکنیم



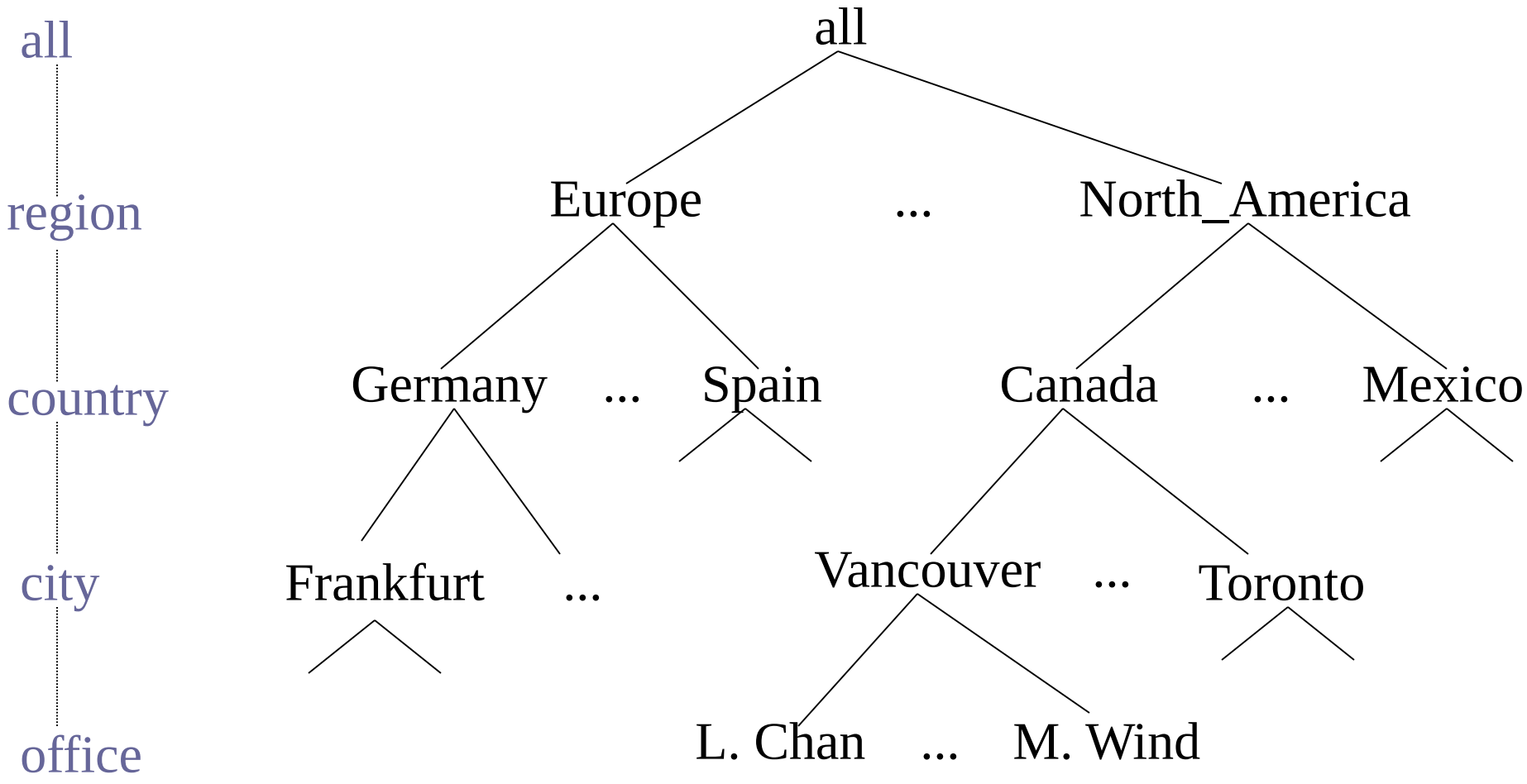
Measures of Data Cube: Three Categories

- Distributive: if the result derived by applying the function to *n* aggregate values is the same as that derived by applying the function on all the data without partitioning
  - E.g., count(), sum(), min(), max()
- Algebraic: if it can be computed by an algebraic function with *M* arguments (where *M* is a bounded integer), each of which is obtained by applying a distributive aggregate function
  - E.g., avg(), min_N(), standard_deviation()
- Holistic: if there is no constant bound on the storage size needed to describe a subaggregate.
  - E.g., median(), mode(), rank()

-
دیتامارت: مجموعه ای از فکت ها و dimensionهای متصل بهش که یک بخش از سازمان ما رو داره کاور میکنه بهش میگیم دیتامارت
گاهی اوقات این aggregation می تونه روی داده هایی که نیاز هست پخش بشه و اینجوری aggregation رو به دست بیاریم ولی بعضی از این aggregation ها هم نمیشه باهاشون کاری کرد و باید به صورت تجمیع همه داده ها رو داشته باشیم که بتونیم aggregation انجام بدیم
: Distributive
مثلا 1 میلیون عدد داریم و می خوام براش sum بگیریم حالا دو تا راه داریم:
1- 1 میلیون عدد رو باهم جمع بکنیم
2- فرض میکنیم این ماشین اینقدر کشش نداره که بتونه این 1 میلیون رو جمع بزنه پس میاد داده ها رو 5 دسته میکنه و حاصل جمع هر کدوم رو میگیره و بعد کل رو جمع میکنه تا عدد اخر به دست بیاد
اگر با مژرهایی سر و کار داریم که این نوع aggregation میخواد روشون انجام بشه کارمون خیلی پیچیده نیست

: Algebraic
اینجا به یک کمکی هم نیاز داریم مثلا برای avg باید count هم بهش بدیم ینی بگیم این avg چندتا عدد است ولی توی sum اینطوری نبود عددا رو میدادیم و تهش می تونستیم جواب نهایی رو داشته باشیم
: Holistic
مثلا میخوایم مد بگیریم این مثل قبلی ها نمیشه --> کلا با یکی یا دوتا کار نمی تونیم بهش برسیم
مثلا برای رنک باید کل داده رو بگیریم و سورتش بکنیم تا بهش برسیم
اینجا کارمون خیلی سخت است

A Concept Hierarchy: Dimension (location)

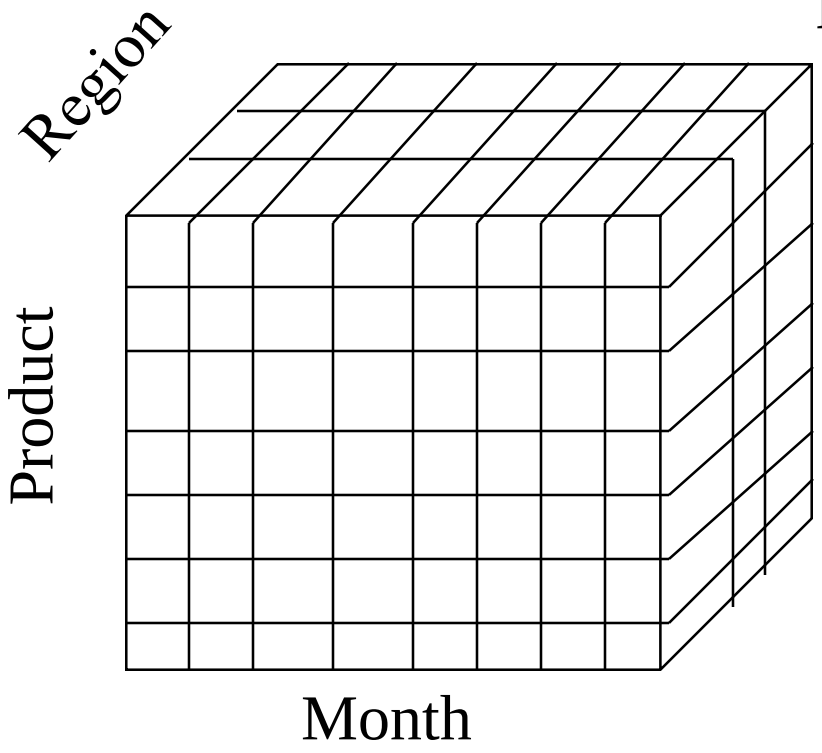


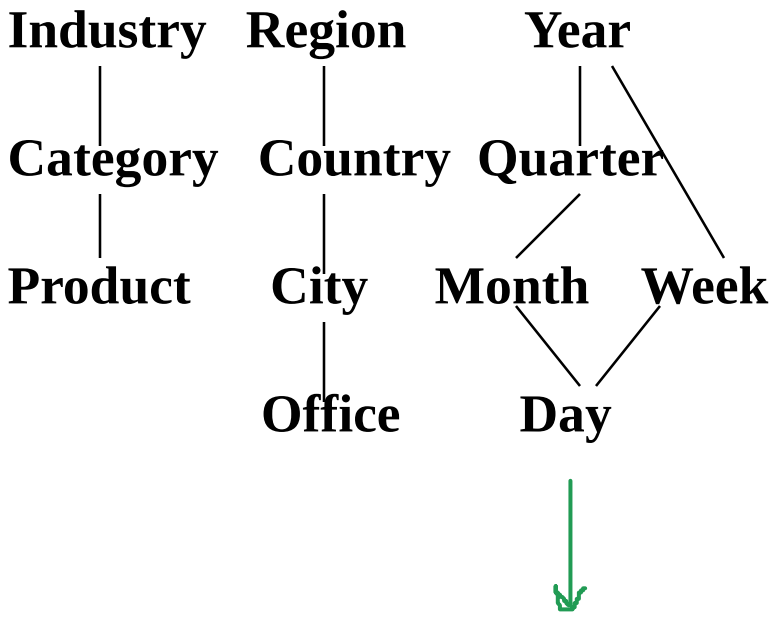
یکی از خاصیت های مهم داخل dimensionها بحث سلسله مراتب است --> ینی یک سلسله مراتب داخل dimension معنا داره
مثلا توی dimension زمان اینارو داریم:
سال و ماه و روز
این سلسله مراتب چه خاصیتی داره؟ وقتی مثلا روی بحث سال نگاه میکنیم این dimension زمان رکوردهای زیادی رو کاور میکنه مثلا وقتی میگیم سال 1390 و اگر میزانگیش درحد روز باشه دراین حالت 365 رکورد dimension رو درگیر میکنه
اگر بعد زمان رو روی سال 90 فیلتر کرده باشیم:
اگر aggregation روی مقدار فروش باشه و sum زده شده باشه، تمام این رکورد ها داره جمع میشه و میگه مقدار فروش در سال 90 چقدر بوده --> اگر این مقدار فروش سال 90 رو بخوایم به تفکیک ماه داشته باشیم برامون برقرار است چون اون 365 تا رکورد موجود در داخل زمان هر کدوم ماهش مشخص است پس به راحتی می تونه شکسته بشه این عدد کلی به 12 ماه ینی مقدار فروش رو برای هر ماه در سال 90 میخوایم ببینیم مثلا 31 روزی که مال فروردین هستن برچسب فروردین 90 میخوره و به همین صورت --> پس یک مقدار میشه الان 12 مقدار
حالا 31 روز فروردین 90 رو می تونیم با جزئیات روز ببینیم
نکته: توی انبار داده باید این خاصیت برقرار باشه ولی اگر از olap engine استفاده بکنیم این سلسله مراتب رو تعریف میکنیم براش
نکته: برای اینکه این سلسله مراتب برای جدول برقرار باشه باید هر فرزند فقط یک پدر داشته باشد و یک فرزند دوتا پدر نداشته باشد مثلا برای ایران پدرش اسیا است و توی رکوردهای دیگه نباید ایرانی رو مشاهده بکنیم که پدرش اسیا نباشه --> شرط اینو توی ETL باید بررسی کنیم و اینم توی STAGING AREA باید بررسی بشه --> توی STAGING AREA باید درستی داده ها رو چک بکنیم
نکته: ممکنه dimensionهایی وجود داشته باشه که سلسله مراتب داخلشون نباش پس حتما اجباری نیست که همه dimensionها سلسله مراتب داخلشون باشه

Multidimensional Data

- Sales volume as a function of product, month, and region

Dimensions: Product, Location, Time
Hierarchical summarization paths





5 تا ستون حتما باید توی جدول داشته باشیم

حجم فروش به عنوان تابعی از محصول، ماه و منطقه

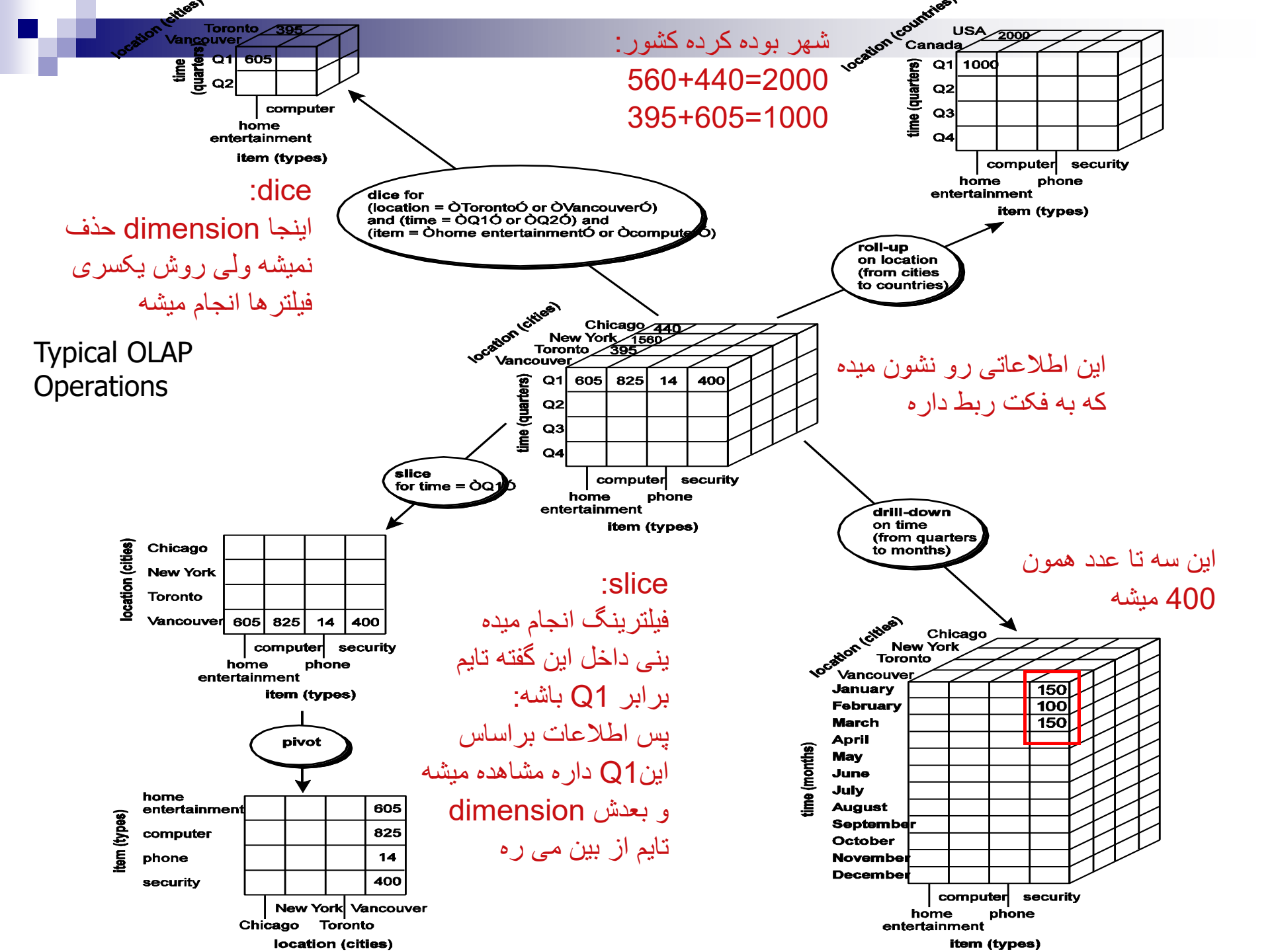
این ینی برای یک dimension می تونیم کوئری های مختفی بزنیم که گزارش های مختلفی بگیریم؟

Typical OLAP Operations

- Roll up (drill-up): summarize data
  - by climbing up hierarchy or by dimension reduction
- Drill down (roll down): reverse of roll-up
  - from higher level summary to lower level summary or detailed data, or introducing new dimensions
- Slice and dice: project and select
- Pivot (rotate):
  - reorient the cube, visualization, 3D to series of 2D planes

Roll up: ینی می خوایم خلاصه تر ببینیم --> مثلا در حد روز داریم می بینیم حالا می خوایم بریم در سطح ماه
Drill down: ینی می خوایم جزئیات رو مشاهده بکنیم --> ینی از سال می خوایم برسیم به ماه
نکته: عملیات Roll up و Drill down بدون سلسله مراتب معنایی نداره

Typical OLAP Operations



مثال:
این یک شکل مفهومی است برای اینکه بهتر درک بکنیم قبلی ها رو
نکته: انبار داده و cube از لحاظ داده ای با هم فرق نمی کنن و فقط از نوع تکنولوژی که پیاده سازی میشه با هم فرق دارن
dimension هایی که داریم لوکیشن و تایم و ایتم است
و اینجا یه دونه مژر داریم که مقدار فروش رو نشون میده --> تلاقی سه dimensionها است
400 ینی توی ایتم security توی ماه اول و توی ونکور چندتا فروش داشته

نکته: توی dice , roll up , drill down برای اینا dimension حذف نمیشه ولی توی slice حذف میشه