پاسخ سری اول تمرین درس پایگاه داده

.1

با توجه به اینکه ممکن است یک سری اطلاعات، به صورت duplicate در چندین فایل موجود باشد، این موجب به افزونگی داده می شود. افزونگی داده نیز، با توجه به اینکه ممکن است همه این داده های duplicate شده با هم آپدیت نشوند، باعث به وجود اومدن ناسازگاری داده می شود.

مورد دیگر ناسازگاری نیز، برآورده نکردن قیود صحت(integrity constraints) در داده ها می باشد. مثلا اگر قرار است که balance های مالی یک تعداد کارمند ذخیره شود، با فرض منفی نشدن balance، سیستم پردازش فایل این قید را تضمین کند و باید برنامه ای نوشت که این قید را برای داده ها بررسی کند.

همچنین، ایجاد یا تغییر دادن قیود صحت نیز در سیستم پردازش فایل دشوار می باشد.

سه مزیت دیگر استفاده از DBMS نسبت به سیستم پردازش فایل به شرح زیر می باشد:

- Atomicity: در DBMS، یا تغییرات لازم به طور کامل انجام می شود و یا اصلا انجام نمی شود. یعنی داده ها در وضعیت سازگاری می مانند و نیمه کاره رها نمی شوند.
- Concurrency: در DBMS، برخلاف سیستم پردازش فایل، امکان دسترسی چندین کاربر با حفظ سازگاری وجود دارد.
- Querying: با آمدن DBMS و ابداع زبان هایی مانند SQL، تغییر دادن داده ها(شامل update کردن، search کردن و delete کردن) راحت تر صورت می گیرد.

.2

مسئولیت کنترل داده و برنامه هایی که به آن داده ها دسترسی دارند به عهده Storage Manager می باشد.

وظایف Storage Manager به شرح زیر می باشد:

- ارتباط با file manager مربوط به سیستم عامل
- ذخیره، بازیابی و بروزرسانی موثر
- ذخیره خود دیتابیس
- ذخیره metadata توصیف کننده ساختار دیتابیس(شمای دیتابیس)

.3

چهار ویژگی اصلی یک تراکنش، به شرح زیر می باشد:

- Atomicity: تراکنش ها باید به صورت اتمیک اجرا شوند؛ یعنی یا به طور کامل با موفقیت انجام شود و یا هیچ کاری انجام نشود.
- Consistency: تراکنش ها نباید باعث شوند که سازگاری داده ها به هم بریزد؛ به عبارت دیگر، بعد از انجام تراکنش، باید قیود صحت(integrity constraints) همچنان برقرار باشد
- Isolation: تراکنش ها باید بتوانند بدون دخالت در کار یکدیگر، انجام شوند. مثلا اگر به طور همزمان وارد شوند، باید به ترتیب اجرا شوند تا روی هم تاثیر نگذارند.
- Durable: تراکنش ها باید تضمین کنند که پس از انجام، تغییراتی که ایجاد کردند، ماندگار می ماند(حتی اگر سیستم crash کند. راه حل این است که یک رکوردی از transaction log ها ذخیره شود تا بتوان هنگام crash کردن، به کمک این رکورد، حالت فعلی سیستم را بازیابی کرد.)

.4

در مدل رویه ای، کاربر در کنار اینکه مشخص می کند که چه داده ای را می خواهد، مشخص می کند که چگونه داده، گرفته شود.

اما در مدل غیررویه ای(توصیفی)، کاربر صرفا مشخص می کند که چه داده ای را می خواهد.

.5

در زیر، چهار تایپ از قیود صحت بررسی شده اند:

- Domain Constraints: برای هر attribute یک tuple، باید مقدار ذخیره شده، عضوی از دامنه ی آن attribute باشد. به عنوان مثال، اگر تنها مقادیر int را می پذیرد(مثلا age ،attribute باشد)، آن گاه مقدار "A" برای آن نادرست خواهد بود.

ID	NAME	SEMENSTER	AGE
1000	Tom	1st	17
1001	Johnson	2nd	24
1002	Leonardo	5th	21
1003	Kate	3rd	19
1004	Morgan	8th	A

Not allowed. Because AGE is an integer attribute

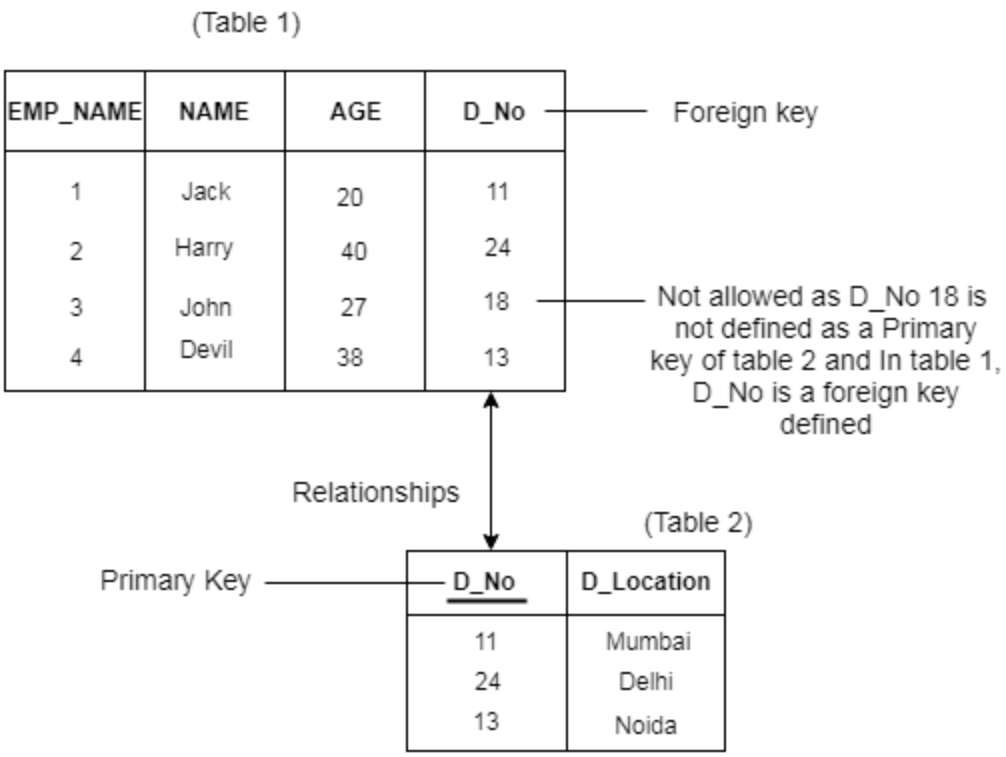
- Entity Integrity Constraints: اگر یک attribute به عنوان primary key معرفی شده باشد، آن گاه نمی تواند مقدار null بپذیرد.(چرا که باید موجب تشخیص tuple ها از هم شود!)

EMPLOYEE

EMP_ID	EMP_NAME	SALARY
123	Jack	30000
142	Harry	60000
164	John	20000
	Jackson	27000

Not allowed as primary key can't contain a NULL value

- Referential Integrity Constraints: اگر foreign key جدول 1، به primary key جدول 2 ارجاع دهد، آن گاه هر مقدار foreign key در جدول 1، یا باید null باشد و یا باید در primary key جدول 2 موجود باشد.(به عبارتی، نباید مقداری در foreign key یک جدول باشد که در primary key متناظر با آن در یک جدول دیگر، موجود نباشد.)



- Key Constraints: مقدارهای primary key، باید بین همه tuple ها، مجزا باشد.

ID	NAME	SEMENSTER	AGE
1000	Tom	1st	17
1001	Johnson	2nd	24
1002	Leonardo	5th	21
1003	Kate	3rd	19
1002	Morgan	8th	22

Not allowed. Because all row must be unique

این قوانین، در data dictionary ذخیره می شوند و به فرم meta data می باشند.

.6

دلیل استفاده از حافظه سلسله مراتبی در دیتابیس(لایه بندی کردن پایگاه داده)، این است که باعث می شود بتوان یک لایه را بدون تغییر دادن بقیه لایه ها، تغییر داد و یا کلا عوض کرد.

لایه های دیتابیس به شرح زیر می باشد:

- Physical Level: پایین ترین لایه می باشد و وظیفه انجام عملیاتی است که توسط کاربر درخواست شده است. تغییر داده واقعی در این لایه صورت می گیرد. همچنین، نحوه ذخیره این داده در این لایه تعریف می شود.
- Logical Level: توصیف می کند که چه داده هایی در دیتابیس ذخیره شده اند.
- View Level: این لایه، وظیفه نمایش دیتابیس(بسته به مدل کاربر) و ارتباط با کاربر را دارد.

.7

جدول می تواند به یکی از صورت های زیر باشد:

- برای کاربر، Attributeهای first name, last name, ID, email, phone number داشته باشد:

First name	Last name	ID	Email	Phone number
Matin	Tavakoli	matin1378	smatintavakolia@gmail.com	**+98 903 720 3775**

که در اینجا، ID و Email و Phone number، هر کدام می توانند super key باشند.

- برای post، Attributeهای name_id, post_id, text, date داشته باشد:

Name_ID	Post_ID	Text	Date
matin1378	#21	“This is a post”	2020/03/06

که در اینجا، post_id می تواند super key می باشد.

همچنین، name_id نیز foreign key می باشد.

- برای comment، Attributeهای name_id, post_id, comment_id, text, date داشته باشد

Name_ID	Post_ID	Comment_ID	Text	Date
matin1378	#21	#7	“This is a comment”	2020/03/06

که در اینجا، comment_id می تواند super key باشد.

همچنین، name_id, post_id نیز foreign key می باشد.

.8

زبان های پرس و جو(یا Querying Languages) به دو دسته زیر تقسیم می شوند:

- Procdeural Query Languages: در این مدل، در کنار این که گفته می شود که چه داده ای مورد نیاز است، نحوه رسیدن به آن داده نیز شرح داده می شود.
  مثال: زبان های برنامه نویسی مانند جاوا
- Non Procedural Query Languages: در این مدل، صرفا گفته می شود که چه داده ای مورد نیاز است و در مورد نحوه رسید به آن داده، چیزی شرح داده نمی شود.
  مثال: SQL(که بر پایه Relational Calculus می باشد.)

.9

الف)

خیر. زیرا ترکیب name و salary لزوما برای هر instructor، منحصر به فرد نیست.

ب)

بله. خیر کلید کاندید نمی باشد؛ زیرا که ID به تنهایی معرف یک tuple منحصر به فرد می باشد و دیگر نیازی به name نداریم.

ج)

کلید ID انتخاب می شود؛ به دو دلیل:

1. عدد می باشد و مقایسه اش راحت است.
2. این attribute دست ما می باشد و جزو مشخصات از قبل تعیین شده فرد نیست.

د)

بله. مثلا می توان برای dept_name موجود در instructor، آن را به dept_name موجود در department ارجاع داد.

.10

1. ممکن است برای یک tuple، آن attribute تعریف نشده باشد.(مثلا اگر جدولی از مشخصات فرد داشته باشیم، ممکن است همه ی tuple ها، attributeی مانند middle name نداشته باشند.)

2. ممکن است برای یک tuple، مقدار آن attribute هنوز مشخص نشده باشد.(مثال ساده: پورتال! اگر بخواهیم یک جدول برای درس های ترمِ دانشجو تعریف کنیم، از همان اول attribute "نمره میان ترم" یا "نمره پایان ترم" آن مشخص نیست. پس تا زمانی که مقدارش مشخص شود، با nul مقداردهی اولیه می شود.)