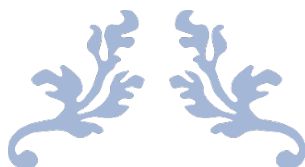

t



جزوه آزمایشگاه پایگاه داده، استاد دیهیم



۶	جلسه چهارم
۶	:ENTITY
۶	اطلاع:
۶	نمودار ER، ENTITY-RELATIONSHIP
۶	درجه ارتباط:
۶	:DB
۶	مجتمع:
۷	به طور اشتراکی و همزمان:
۷	اجزا مدل رابطه‌ای:
۷	:DOMAIN میدان
۷	درجه رابطه:
۷	کاردینالیتی رابطه:
۷	رابطه نرمال:
۷	مزایای میدان:
۸	نکات مهم در پیاده سازی DOMAIN:
۸	P.K (کلید اصلی):
۸	F.K (کلید خارجی):
۸	قواعد جامعیت:
۹	C1(موجودیتی):
۹	C2 (ارجاعی):
۱۰	مهم ترین نرم افزارهای مدیریت پایگاه داده:
۱۰	:SQL SERVER
۱۱	:SQL SERVER نصب
۱۱	:SQL نسخه های
۱۱	:SQL اجرای
۱۲	:SQL امکانات جدید
۱۳	جلسه پنجم

۱۳	مدیریت پایگاه داده:
۱۳	ساخت پایگاه داده:
۱۳	ساخت جدول:
۱۴	:DATA TYPES
۱۴	ذخیره جدول:
۱۴	ویرایش جدول:
۱۵	افزودن فیلد:
۱۵	حذف فیلد:
۱۵	وارد کردن رکوردهای جدول:
۱۵	ثبت رکورد:
۱۵	حذف رکورد:
۱۵	حذف جدول:

جلسه ششم

۱۵	محدودیت CONSTRAINT
۱۵	پیاده سازی محدودیت ها:
۱۶	نرمال سازی:
۱۶	مشخصات پایگاه داده نرمال:
۱۷	سطوح نرمال سازی:
۱۸	انواع رابطه:
۱۸	پیاده سازی روابط با DIAGRAM:

جلسه هفتم

۲۰	1- INSERT:
۲۱	2- UPDATE:
۲۱	3- DELETE:
۲۱	4- SELECT:
۲۲	5- ORDER BY

۲۳	6- TOP
۲۳	7- BETWEEN
۲۳	:JOIN
۲۴	انواع JOIN:

۲۶ جلسه هشتم

۲۶	1- تابع MAX
۲۷	2- تابع MIN
۲۷	3- تابع COUNT
۲۸	4- تابع SUM
۲۸	5- تابع AVG
۲۹	کوئری‌های تو در تو:
۲۹	1- IN, EXIST
۳۰	2- NOT IN, NOT EXISTS
۳۱	3- DISTINCT
۳۱	4- SOME, ANY
۳۱	5- ALL
۳۲	6- HAVING

۳۳ جلسه نهم

۳۳	:VIEW
۳۶	ویرایش VIEW
۳۶	محدودیت‌های VIEW:
۳۷	:SP: STORED PROCEDURE
۳۷	قابلیت‌های SP:

۴۰ جلسه دهم

۴۰	پارامترهای اختیاری در SP:
----	---------------------------

ویرایش SP: ۴۲

فصل ۶، محافظت از پایگاه داده ۴۲

۴۲ : BACKUP

۴۳ :RESTORE

۴۳ :DETACH

۴۳ :ATTACH

۴۴ :DTS: DATA TRANSFORMATION SERVICES

۴۴ :IMPORT

۴۵ :EXPORT DATA

جلسه نازدهم ۴۶

فصل ۷، امنیت پایگاه داده. ۴۶

نقش‌های پیش فرض SQL: ۴۶

۴۶ :BULK ADMIN

۴۶ :DB CREATOR

۴۶ :DISK ADMIN

۴۶ :PUBLIC

۴۶ :PROCESS ADMIN

۴۷ :SECURITY ADMIN

۴۷ :SERVER ADMIN

۴۷ :SYS ADMIN

۴۷ :SETUP ADMIN

تعریف حساب کاربری جدید: ۴۷

فصل ۸، اتصال برنامه به پایگاه داده ۴۸

روش‌های برقراری ارتباط بین برنامه و پایگاه داده: ۴۸

۴۸	مدل سازی:
۴۹	مدل سازی با EF:

۴۹	جلسه ۱۲، ساخت پروژه (جلسه آخر)
----	--------------------------------

۴۹	MDI FORM:
۴۹	افزودن فرم MDI به پروژه:
۵۰	حذف کنترل های پیش فرض MDI:
۵۰	حذف کدهای کنترل های حذف شده:
۵۰	تعیین فرم اولیه برنامه جهت اجرا:
۵۰	ثبت اطلاعات:
۵۲	ایجاد منو برای فرم اصلی
۵۳	ایجاد زیر منو:
۵۳	وارد کردن اطلاعات:
۵۴	بررسی درستی ثبت اطلاعات:
۵۴	نمایش LIST:
۵۵	حذف اطلاعات:
۵۵	بروزرسانی محتوای GRIDVIEW:

SQL: Structure Query Language

:Entity

هر چیز که می‌خواهیم در مورد آن اطلاع داشته باشیم، شامل مجموعه‌ای از صفات یا attribute است.

اطلاع:

معنایی است که انسان به Data می‌دهد.

نمودار ER، Entity-Relationship

نمودار موجودیت ها و ارتباطات است. نمایانگر ارتباط بین موجودیتها.

درجه ارتباط:

1:1, 1:n, m:n

:DB

مجموعه‌ای از داده‌های ذخیره شده در مورد انواع موجودیت‌های یک محیط عملیاتی و ارتباطات بین آنها به صورت مجتمع و مبتنی بر یک ساختار تعریف شده به طور صوری با حداقل افزونگی تحت کنترل متمرکز، مورد استفاده یک یا چند کاربر به طور اشتراکی و همزمان.

مجتمع:

کل داده‌ها در کادر یک ساختار مشخص به شکل یکجا ذخیره می‌شوند.

تعریف شده به طور صوری:

داده‌ها به کمک زبان خاصی تعریف می‌شوند سیستم باید به کاربران اجازه بدهد تا داده هایشان را به دور از جنبه‌های پیاده سازی و ذخیره سازی تنها به آن شکل که خود می‌بیند تعریف کنند.

حداقل افزونگی:

مقادیر یک یا چند صفت یکجا ذخیره نشود.

به طور اشتراکی و همزمان:

هر کاربر بدون ایجاد محدودیتی برای دیگر کاربران بتواند از DB استفاده کند. همزمانی فرایندها
به نحوه طراحی DBMS و الگوریتم‌های عملیاتی و عملکرد OS و معماری کامپیوتر وابسته است.

اجزا مدل رابطه‌ای:

- جدول
- ستون (صفات)
- سطر: نمونه موجودیت

میدان Domain:

مقادیر مجاز هر صفت را گویند

درجه رابطه:

تعداد صفات رابطه

کار دینالیتی رابطه:

تعداد تاپلهای رابطه در هر زمان

رابطه نرمال:

تمام مقادیر صفات آن اتومیک است یعنی در تلاقی هر سطر و ستون تنها یک مقدار ساده تجزیه
ناپذیر وجود دارد.

مزایای میدان:

۱. Query ها را کنترل میکند

Create Domain sid char(5);

نمی‌توانیم برای sid بیش از ۵ کارکتر وارد کنیم.

۲. Query ها را از لحاظ Semantic کنترل می‌کند

نمی‌توانیم Query داشته باشیم که مثلا درخواست کند وزن قطعه برابر تعداد تهیه شده
از قطعه دیگر باشد زیرا وزن و تعداد دو میدان متفاوت‌اند.

۳. پاسخگویی برخی Query ها را آسان می کند.

فرض کنید این Query مطرح شده: در چه رابطه هایی حداقل یک صفت وجود دارد؟
Query: در چه رابطه هایی حداقل یک صفت وجود دارد که روی میدان شماره تهیه کننده تعریف شده باشد؟

نکات مهم در پیاده سازی Domain:

۱. حاصل هرگونه عملیات روی میدان باید بسته باشد.
۲. سیستم باید عملگرهای تک عملوندی روی هر میدان را مشخص کند.
۳. سیستم باید مشخص کند برای هر دو میدان چه عملگرهای دو عملوندی قابل اعمال است
۴. سیستم باید مشخص کند برای هر عبارت عملیاتی میدان نتیجه عبارت چه خواهد بود.

P.K (کلید اصلی):

هر زیرمجموعه از صفات دارای دو ویژگی زیر باشد:

۱. یکتایی مقدار Unique
 ۲. Minimality (با حذف هر یک از صفات این زیرمجموعه یکتایی از بین برود)
- ✚ هر رابطه یک P.K دارد که در بدترین حالت مجموعه کل صفات است.
- ✚ اگر چند کاندید برای کلید اصلی وجود داشت آنی باید انتخاب شود که از نظر طول کوتاهتر باشد و مهمترین زیر مجموعه صفات را داشته باشد.

F.K (کلید خارجی):

صفت A1 از رابطه R2 یک کلید خارجی است اگر A1 در رابطه R1 (نه لزوما متمایز) P.K باشد.

✚ کلید خارجی امکانی است برای پیوند دادن رابطه های پایگاه داده با هم ولی تنها امکان ایجاد ارتباط نیست. هر صفت مشترک بین دو رابطه عاملی است برای نمایش ارتباط بین رابطه ها.

قواعد جامعیت:

۱. عام

۱/۱. C1: موجودیتی

۱/۲. C2: ارجاعی

۲. خاص

۲/۱. میدانی

۲/۲. صفتی

۲/۳. رابطه‌ای

۲/۴. پایگاهی

C1 (موجودیتی):

هیچ جز تشکیل دهنده کلید اصلی نمیتواند Null باشد.

C2 (ارجاعی):

اگر صفت A1 در R2 کلید خارجی باشد:

- A1 می‌تواند Null باشد به شرط آنکه جز کلید اصلی نباشد.

- اگر Null نبود حتما باید مقدار یکسانی در R1 داشته باشد.

✚ برای عملیات درج و بهنگام سازی سیستم از روی گراف ارجاع کنترل لازم برای اعمال C2 را انجام می‌دهد.

برای عمل حذف اعمال C2 روش های زیر وجود دارد:

۱- Cascade (انتشاری)، با حذف تاپل مرجع تاپلهای رجوع کننده نیز حذف می‌شوند

۲- Restricted (تعویقی)، حذف تاپل مرجع تا زمانیکه تاپلهای رجوع کننده وجود دارد به تعویق افتند.

۳- Nullifying (هیچ مقدار گذاری)، با حذف تاپل مرجع، کلید خارجی در تاپل‌های رجوع Null شود به شرط آنکه جز کلید اصلی نباشد.

۴- مقدار گذاری با مقدار پیشفرض، به جای Null یک مقدار Default قرار گیرد.

۵- عدم اقدام، کلا درخواست رد شود.

اعمال قواعد خاص:

Constraint Check

Assertion ... Check

Create trigger ... for

مهمترین نرم افزارهای مدیریت پایگاه داده:

۱. مایکروسافت

۱/۱. Access: تحت win در مقیاس کوچک

۱/۲. SQL Server: تحت win و web و شبکه با زبان برنامه نویسی C#

۲. شرکت Oracle

۲/۱. Oracle: پرستفاده ترین Java-DB تحت win و web و شبکه

۲/۲. My SQL: پایگاه داده وب سایت های اینترنتی با PHP

SQL Server:

امکان تعریف چند پایگاه داده و قابلیت اجرای چند پایگاه داده

پایگاه داده های سیستمی و پیش فرض SQL:

۱- Master: موجود در تمام نسخه ها حاوی جداولی برای ردیابی فعالیت های انجام شده

ضروری و غیر قابل حذف.

۲- Model:

- پایگاه داده ای حاوی قوانین و استانداردهای ساخت DB

- افزودن مقداری از جداول پر استفاده در تمامی پایگاه داده هایی که ایجاد می شوند

- افزودن مقداری از کاربران و گروه های کاربردی به طور خودکار به پایگاه داده ایجاد شده.

۳- MSDB: ذخیره فعالیت های SQL Agent، تنظیماتی برای تهیه روزانه نسخه پشتیبان

۴- Temp DB: مهم‌ترین عنصر سرور حاوی کوئری‌های موقت و جداول موقت (تمام

داده‌های موقت)

پایگاه داده‌های دیگر برخی SQLها:

۱- Report Server: تنظیمات مربوط به گزارش و ذخیره مدل‌ها (درخواست یک شماره پیش فیش با الگویی خاص تولید شود و در آغاز هر روز شماره فیش‌ها از ابتدا تولید شوند)

(جزوه ارسالی جلسه چهار در گروه از پس از صفحه ۹، صفحه ۱۲ قرار گرفته و صفحه ۱۰ و ۱۱ ثبت نگردیده)

نصب SQL Server:

در نسخه ۲۰۱۷ بایستی نرم افزارهای زیر که به صورت جدا از هم می‌باشند نصب شود:

۱- Data Base Engine

۲- Managements Studio

نسخه‌های SQL:

۱- Express: کم حجم مناسب برنامه نویسان وب

۲- Standard Edition: مناسب برای سازمان‌های کوچک

۳- Business Edition: مناسب باری سازمان‌های متوسط

۴- Web & Developer Edition: نسخه کامل، برای اهداف و محصولات تجاری نمی‌تواند

استفاده شود.

۵- Enterprise Edition: کامل‌ترین نسخه، از حداکثر پردازنده، حافظه رم بهره می‌برد ایضا

پشتیبانی تمام امکانات پیشرفته

اجرای SQL:

➔ Microsoft SQL Server Management Studio

➔ Server type: Database Engine

Server name: 0 →

Authentication: windows →

نوع سرور به سیستمی که می‌خواهید به آن وارد شوید مربوط می‌شود. با انتخاب DB Engine موتور SQL راه اندازی می‌شود.

اگر هنگام نصب SQL نام System را به عنوان نام سرور انتخاب کرده باشید هنگام ورود برای نام سرور نام کامپیوتر خود را باید وارد کنید، تایپ نقطه (dot) به این معناست که شما تمامی نامهای سرور موجود بر روی سیستم را انتخاب کرده‌اید.

برای Authentication دوگزینه دیگر نیز وجود دارد:

۱- Windows یعنی پایگاه داده روی کامپیوتر شخصی کار می‌کند.

۲- Server یعنی پایگاه داده روی شبکه یا وب کار می‌کند پس نام کاربری و رمز عبور نیاز دارد.

امکانات جدید SQL:

- در لینوکس نیز قابل استفاده است
- بهینه سازی کوئری‌ها با توجه به زمان بارگذاری برنامه
- ادامه دادن به عملیات بازسازی ایندکس‌های آنلاین از جایی که متوقف شده
- ارائه مشکلات احتمالی عملکردهای کوئری و راه حل آنها
- برطرف کردن مشکلات به طور خودکار
- روابط گرافیکی یکپارچه شده
- عدم نیاز به استفاده از ویندوز کلاستر برای پیکربندی AOAG بین ویندوز و سرورهای لینوکس

با اجرای SQL پنجره Explorer Object باز می‌شود. اگر باز نشد، F8 بزنید

جلسه پنجم

مدیریت پایگاه داده:

OE (Object Explorer) -> + Data Bases

ساخت پایگاه داده:

۱- OE -> Right click on DB -> new DB

۲- Choose your name: DB Name

نام باید انگلیسی باشد و قاعده PascalCase بدون کارکتر خاص و فاصله – کلمه کلیدی نباشد.

۳- تعیین مالک پایگاه داده:

Owner -> ... -> Browse -> [SA]

مدیر سیستم

۴- DB files:

الف) فایل اصلی (.mdf): File Type: Rows Data

ب) فایل Log (.ldf): حاوی تمام فعل و انفعالات انجام شده در پایگاه داده که برای

خطایابی قابل استفاده است

File Type: LOG

Initial Size: فضای اختصاص داده شده به هر فایل

Auto growth/Max size: حجم فایلها در صورت رشد تا چه اندازه افزایش یابد و max آن

محدود باشد یا نامحدود.

✚ برای مشاهده پایگاه داده و عناصر آن روی علامت + کنار نام DB کلیک می کنیم.

ساخت جدول:

+DB -> click right on Tables -> New Table

۱- Column Name

۲- Data Type

۳- Allow Nulls

Data Types

- Bit, TinyInt: 1B
- SmallInt: 2B
- Int, SmallMoney: 4B
- BigInt, Money, Date/Time: 8B
- UniqueIdentifier: 16B
- Char, Nchar: ثابت
- متغیر: Numeric or Decimal, Float, varchar, NVarchar

NChar(n): کارکترهای غیر لاتین که طولی حداکثر n دارند

NVarchar(n): رشته‌ای از کارکترهای لاتین با طول حداکثر n که اگر طول رشته کوتاه‌تر شد فضای اضافه آزاد شود.

فقط فیلدهایی را از نوع عددی انتخاب کنید که می‌خواهید روی آنها عملیات ریاضی انجام دهید. بنابراین کدملی، شماره تلفن شماره شناسنامه، کد دانشجویی از نوع عددی نیست.

زیرا میانگین سن و مجموع سن معنا ندارد. Age -> TinyInt

زیرا محاسبه ندارد، فارسی نیست و طولش ثابت است char -> Phone

برای فیلدهای ضروری که حتما باید با مقداری پر شود گزینه Allow Nulls تیک نمی‌خورد مثل نام، تلفن و ...

ذخیره جدول:

کلیک روی علامت دیسکت یا ctrl + s پس تعیین نام جدول

بستن پنجره طراحی جدول:

کلیک روی علامت X بالای پنجره

ویرایش جدول:

راست کلیک روی جدول -> design

افزودن فیلد:

Right click on the Table -> Design Insert Column

حذف فیلد:

Right click on field -> Delete Column

وارد کردن رکوردهای جدول:

OE -> click right on Table -> Edit top 200 rows

ثبت رکورد:

بعد از وارد کردن اطلاعات یک رکورد دکمه tab را فشار دهید اگر اطلاعات از نظر نوع و ساختار درست و وارد شده باشد رکورد ثبت می شود.

حذف رکورد:

سمت چپ رکورد مرود نظر کلیک کنید تا رکورد انتخاب شود سپس کلیک راست کرده Delete را انتخاب کنید.

حذف جدول:

OE -> Tables -> click right on table -> Delete

جلسه ششم

محدودیت Constraint

- ۱- دامنه‌ای: روی یک یا چند ستون اعمال می شود
مثال: از ستون سن فقط اعداد مثبت وارد شده اند
- ۲- موجودیتی: مثل یکتا بودن کد ملی
- ۳- ارجاعی: مقدار کلید خارجی باید با کلید اصلی متناظرش یکسان باشد.

پیاده سازی محدودیت ها:

۱- یکتایی: Unique

۲- بررسی: Check

شرط گذاری روی ورود اطلاعات یک یا چند ستون:

- مثلا: سن بین ۱۸ الی ۶۰ باشد

- مثلا: تاریخ خرید => تاریخ سفارش

- مثلا: وضعیت تاهل: مجرد - متاهل

۳- پیش فرض Default:

اگر مقداری برای فیلد مورد نظر وارد نشود با مقدار پیش فرض پر شود مثلا پیش فرض وضعیت تاهل مجرد باشد.

نرمال سازی:

فرض کنید می خواهیم برای یک فروشگاه موبایل پایگاه داده ایجاد کنیم. فرض کنید جدول زیر را طراحی کرده ایم:

(شماره فاکتور، تاریخ، نام مشتری، آدرس، سفارشات)

بزودی متوجه خواهیم شد که مشکلات ریز و درشتی گریبان گیر ما خواهد شد:

۱- برای مشتری اطلاعات بیش تری نیاز داریم: مثلا شماره تلفن و ...

۲- برای محصول هم با کمبود اطلاعات مواجه ایم

اگر این فیلدها نیز اضافه شوند و خریدار چند محصول خریداری کند مشخصات او مرتبا تکرار می شود.

اگر از یک محصول چند عدد خریداری شود مشخصات محصول نیز چند بار تکرار می شود که در نتیجه آن افزونگی اطلاعات داریم.

افزونگی باعث افزایش حجم اطلاعات ذخیره شده و کاهش سرعت نیز منجر می شود.

راه حل: RDB: تفکیک اطلاعات به چند جدول در ایجاد رابطه بین آنها برای استخراج اطلاعات.

مشخصات پایگاه داده نرمال:

۱- هر جدول فقط مختص یک موجودیت است

۲- کلید اصلی دارد

۳- ترتیب سطر و ستون تاثیری در عملکرد پایگاه داده ندارد.

سطوح نرمال سازی:

۱- سطح اول: هر موجودیت یک جدول با کلید اصلی دارد

جدول سفارش (شماره فاکتور، تاریخ، کد مشتری، سفارشات)

جدول مشتری (کد مشتری، نام مشتری، آدرس)

* حتما باید فیلد مشترک بین جدول وجود داشته باشد تا از طریق آن بتوان اطلاعات مرتبط را استخراج کرد.

* کد مشتری در یکی کلید اصلی و در دیگری کلید خارجی به حساب می آید.

* در ستون سفارشات بیش از یک سفارش ثبت شده، فقط نام محصول قید شده، باید تعداد و قیمت نیز مشخص شود.

سفارش (شماره فاکتور، تاریخ، کد مشتری، محصول، رنگ، تعداد، قیمت، قیمت کل) ایراد داره چون کلید اصلی ندارد.

۲- سطح دوم:

هدف کاهش داده های تکراری (افزونگی) است.

-> افزودن ستون و ردیف: تا ردیف و شماره فاکتور با هم کلید اصلی شوند.

-> برای اطلاعات محصول افزونگی وجود دارد -> تجزیه محصول

* جدول Header حاوی اطلاعات پایه ای و اصلی

* جدول جزئیات حاوی سایر اطلاعات

هر رکورد از جدول header به تنها یک رکورد از جدول جزئیات اشاره می کند ولی هر

رکورد از جدول جزئیات می تواند به چند رکورد از جدول header اشاره کند -> رابطه

1:n

جدول سفارش (شماره فاکتور، تاریخ، کد مشتری)

جزئیات سفارش (شماره فاکتور، ردیف، محصول، رنگ، تعداد، قیمت، قیمت کل)

۳- سطح سوم:

هیچ فیلدی نباید به فیلدی غیر از کلید اصلی وابسته باشد.
فیلدهای قیمت و رنگ وابسته به محصول اند -> اطلاعات محصول باید در جدول دیگر قرار گیرد.

محصول (کد محصول، نام محصول، رنگ، قیمت)
سفارش (شماره فاکتور، تاریخ، کد محصول، کد مشتری، تعداد)
* نباید فیلد ترکیبی داشته باشیم، نام و نام خانوادگی باید از هم جدا باشد
مشتری (کد، نام، نام خانوادگی، آدرس)

انواع رابطه:

- یک به یک ۱:۱، دانشجو- حساب کاربری
 - یک به ان 1:n، شرکت - محصول
 - ام به ان M:n: دانشجو - درس
- * برای بررسی رابطه m:n یک جدول مجزا در نظر گرفته می شود.

پیاده سازی روابط با Diagram:

- ۱- پایگاه داده ای به نام MyStore ایجاد کنید
- ۲- جدول customer به شکل زیر ایجاد شود.

Customer (CustomerId, FName, LName, Phone, Adr)

تعیین کلید اصلی:

روی فیلد مورد نظر کلیک راست و گزینه Set as primary key را انتخاب می کنیم.
کنار آن علامت کلید ظاهر می شود و تیک گزینه Allow Null به طور خودکار خاموش می گردد.

مقدار دهی خودکار کلید اصلی توسط سیستم:

کلیک روی فیلد -> Column Properties -> Is Identity: Yes -> شناسه چند تا چند تا اضافه شود Identity Increased -> Identity seed

۳- کلید اصلی را تعیین کرده و برای آن مقدار اولیه ۱۰۰۱ را منظور کنید.

۴- جدول product را بسازید

Product(ProductId, ProductName, Color)

* برای کلید اصلی مقدار اولیه ۲۰۰ را تعیین کنید

۵- جدول سفارش Order را بسازید:

Order (OrderId, ProductId, CustomerId, Date, Count)

* فیلدهای کلید خارجی حتما باید هم نوع با کلید اصلی مربوطه باشند.

* کلید خارجی می تواند نامی متفاوت با کلید اصلی مربوطه داشته باشد

۶- ایجاد رابطه در جدول:

My Store <- DB Diagram کلیک راست <- New Diagram <- پیام: پایگاه

داده اشیا مورد نیاز را ندارد آیا می خواهید آن را ایجاد کنید؟ "بله" <- مشاهده نام جداول

<- با عمل Drag تمام جدول را انتخاب کنید <- مشاهده نام جداول <- add <-

close

* فیلد مشترک بین دو جدول را اینگونه به هم وصل کنید که کلید خارجی را با drag به

کلید اصلی متناظر وصل کنید <- پنجرهای باز می شود <- OK <- Ok

جدول سمت ۱ <- جدول ۱

جدول سمت n <- جدول سمت بینهایت

۷- Save:

* دیالگرم را با ctrl + S ذخیره کنید.

۸- Update:

* پایگاه داده انتخاب را کرده روی علامت Refresh کلیک کنید تا بهنگام سازی شود

جلسه هفتم

رشته‌ها داخل 'Single Quotation' قرار می‌گیرند نه "Double Quotations" توجه شود!

فصل سوم، دستورات T-SQL

۱- Insert:

درج اطلاعات در جدول

```
Insert Into <Table Name> (F1, F2..., Fn)  
Values (V1, V2..., Vn)
```

```
Insert Into Product (ProductName, Color, Price)  
Values ('Galaxy S9', 'Gold', 500)
```

* برای ثبت درخواست روی New Query کلیک کنید

* مقدار رشته درون " قرار می‌گیرد.

* برای مقادیر فارسی (یونیکد) در پایگاه داده قبل از آن حرف N را قرار دهید.

* برای اجرای دستور روی دکمه Execute کلیک کنید

* از آنجا که کد مشتری و کد محصول به شکل خودکار تولید می‌شود برای وارد کردن اطلاعات در جدول سفارش باید کدهای معتبر و درستی وارد شود.

```
Insert Into [Order] (CustomerId, ProductId, Date, Count)  
Values (1001, 200, '1399/01/01', 10)
```

* کلمه order در SQL کلمه کلیدی است بنابراین باید نام جدول را در [] قرار دهیم تا مشکلی رخ ندهد.

۲- Update:

ویرایش اطلاعات وارد شده

مقدار ۱ = فیلد ۱ set نام جدول Update

مقدار ۲ = فیلد ۲

شرط where

Ex:

Update Customer Set Adr = 'N' 'ولیعصر'

Where CustomerId = 1001

۳- Delete:

نام جدول Delete From

شرط Where

حذف یک یا چند رکورد

EX:

Delete From [Order] where orderId = 109

* دقت کنید که اگر شرط گذاشته نشود تمامی سفارشات حذف می شود.

۴- Select:

استخراج اطلاعات

نام جدول From ... و فیلد ۲ و فیلد ۱ Select

شرط Where

* برای استخراج تمام فیلدهای جدول از علامت * استفاده می کنیم

* وجود شرط در این دستور اختیاری است.

Select * From Customer

نمایش اطلاعات مشتریان

Select ProductName, Price From Product

نمایش نام و قیمت تمام محصولات

Select * From Customer

where LName = N'مهدوی'

* در SQL از عملگرهای زیر می‌توان استفاده کرد:

= < > (مخالف) AND OR

کوئری: نمایش لیست محصولاتی که قیمت آنها بیش‌تر از ۱۰۰۰ دلار باشد:

Select * From Product

Where Price > 1000

کوئری: نمایش محصولاتی که رنگ آنها قرمز یا نقره‌ای است:

Select * From Product

Where Color = 'red', Or Color = N"نقره‌ای"

کوئری: نمایش محصولاتی که رنگ آنها طلایی و قیمت آن کم‌تر از ۵۰۰ باشد:

Select * From Product

Where color = 'red' And Price < 500

۵- Order By

نمایش مرتب شده اطلاعات

Select * From <Table Name>

order by <Field Name> [asc | desc]

asc صعودی

desc نزولی

Select * From [Order]
Order By Count asc

نمایش اطلاعات سفارش براساس ترتیب صعودی تعداد سفارش

۶- Top

استخراج N عنصر اول یک جدول به ترتیب درج رکوردها

Select Top n * From نام جدول

Ex:

Select Top 2 * From Product
Order By Price desc

۷- Between

شرط بین دو مقدار است

Select * From [Order]
Where Count Between 2 and 4

:Join

فرض کنید کوئری‌های زیر مطرح شده است:

کوئری: تعداد سفارش‌های هر محصول به همراه مشخصات سفارش دهنده

کوئری: تعداد فروخته شده از هر محصول به همراه تاریخ فروش

برای کاهش افزونگی جداول را تفکیک کردیم. اما اکنون برای یکپارچه کردن اطلاعات و پاسخ به کوئری‌های فوق باید جداول را ادغام نماییم، برای اینکار از Join استفاده می‌کنیم. Join باعث می‌شود که فیلدها در کنار هم نمایش داده شوند.

انواع Join:

1 - Inner Join

اطلاعات مشترک هر دو جدول نمایش داده می‌شود.

کوئری: به دست آوردن تعداد فروخته شده هر کالا

-> کد محصول، نام محصول، تعداد

-> جدول: Product (نام و کد) و Order (کد و تعداد)

Select Table1, field1, Table2, field2, ... From Table1 Inner Join

Table 2 on Table1.PK = Table2.Fk

* ادغام جداول از طریق فیلد مشترک صورت می‌گیرد

اسم مستعار:

برای کوتاه‌تر شدن دستورات از نام مستعار برای جداول یا فیلدها استفاده می‌شود. کافی

است هر جا برای اولین بار نام جدول آمد بنویسیم:

نام مستعار AS نام جدول

کوئری: گزارشی تهیه کنید که حاوی نام، نام خانوادگی، تعداد، نام محصول، قیمت و قیمت کل باشد.

Select C.FName, C.LName, P.Product Name, P.Price, O.Count

As Total From Customer

As C Inner Join [order] As O on

C.CustomerId = O.CustomerId

Inner Join Product As P on

P.ProductId = O.ProductId

* برای فیلد مشتق که اطلاعاتش در جدول موجود نیست نیز باید یک نام مستعار

سفارش دهیم مثل Total

Left Join & Right Join - ۲

کوئری: از هر محصول چه تعداد سفارش داده شده؟

-> کد محصول، نام محصول، تعداد سفارش

-> product, order

اطلاعات محصولاتی که سفارش داده نشده هم نمایش داده می شود

Inner Join پاسخگو نیست

a. Left Join

تمام رکوردهای جدول سمت چپ نمایش داده می شود اگر به ازای آنها در جدول

سمت راست مقداری بود نمایش داده شود و اگر مقداری نبود Null گذاشته شود.

```
Select P.ProductName, O.Count
```

```
From Prodct As P Left Join
```

```
[Order] As O on P.ProductId = O.ProductId
```

b. Right Join

کوئری: مشخصات مشتریان به همراه سفارشات آنها:

```
Select * From [Order] As O
```

```
Right Join Customer As C
```

```
on O.CustomerId = C.CustomerId
```

* اگر جای جدول عوض شود نوع Join نیز عوض می شود. (Right به Left تغییر

می کند)

c. Full Join

وقتی نیاز است تمام اطلاعات دو جدول چه مشترک و چه نا مشترک در Join

گنجانده شود از Full Join استفاده می کنیم.

کوئری: نام کامل مشتریانی که سفارش داده اند و سفارش نداده اند و نام قیمت

محصولاتی که سفارش داده شده/ند و سفارش داده نشده/ند

```
Select C.FName, C.LName, P.ProductName, P.Price  
From Customer As C Full Join  
[Order] As O On  
C.CustomerId = O.CustomerId  
Full Join Product As P On  
P.ProductId = O.ProductId
```

Cross Join .d

دو جدول را در هم ضرب می‌کند یعنی به ازای هر رکورد جدول اول تمام رکوردهای جدول دوم را پیوند می‌دهد

```
Select * From Customer  
As C Cross Join [Order] As O
```

جلسه هشتم

فصل ۴، توابع پرکاربرد SQL

۱ - تابع Max

برای محاسبه Max/استفاده می‌شود و
* باید برای نتیجه نامی انتخاب کرد.

```
Select Max (نام فیلد) As نام نتیجه  
From <Table Name>
```

کوئری: بالاترین قیمت محصول را بدست آورید.

```
Select Max (Price) As MaxPrice  
From Product
```

کوئری: بیشترین تعداد فروخته شده را بدست آورید،

```
Select Max (Count) As MaxCount  
From [Order]
```

۲ - تابع Min

برای محاسبه Min/استفاده می شود

*برای نتیجه نامی انتخاب کنید.

کوئری: محصول با کمترین قیمت را بیابید:

```
Select Min (Price) As MinPrice  
From Product
```

کوئری: کمترین قیمت فروش را به دست آورید:

```
Select Min (Count) As MinCount  
From [Order]
```

۳ - تابع Count

برای محاسبه تعداد استفاده می شود.

کوئری: تعداد مشتریان را بدست آورید:

```
Select Count (CustomerId) As CountOfCustomer  
From Customer
```

کوئری: تعداد محصولات را بدست بیاورید:

```
Select count (ProdcutId) As CountOfProduct  
From Product
```

۴- تابع Sum

برای محاسبه مجموع به کار می‌رود
* به همراه دستور Group By استفاده می‌شود.
کوئری: مجموع فروش هر محصول را بدست آورید:
اول: فروش هر محصول را بدست می‌آوریم (نام محصول، تعداد سفارش)
دوم: تعداد فروش محصولات یکسان جمع می‌شود.

```
Select P.ProductName,  
Sum(O.Count) As sumOfCount  
From product As P Inner Join  
[Order] As O on  
P.ProductId = O.ProductId  
Group By P.ProductName
```

کوئری: دو محصول پر فروش را بدست آورید (تعداد فروش = سفارش، نام محصول)

۱- نام محصول و تعداد فروش <= Inner Join

۲- محاسبه مجموع فروش هر محصول <= Sum Group By

۳- مرتب کردن نزولی نتیجه قبل و برگرداندن دو رکورد اول نتیجه <= Top Order By

```
Select Top 2 P.ProductName,  
Sum(O.Count) As sumOfCount  
from Product As P Inner Join  
[Order] As O on  
P.ProductId = O.ProductId  
Group By P.ProductName  
Order By SumOfCount desc
```

۵- تابع AVG

برای محاسبه میانگین استفاده می‌شود
با دستور Group By به کار می‌رود

کوئری: میانگین قیمت محصولات:

```
Select AVG (Price) As AvgOfPrice  
From Product
```

کوئری: میانگین تعداد فروش هر محصول:

۱- از هر محصول چند مورد فروخته شده، Inner Join

۲- میانگین تعداد فروش هر محصول: AVG, Group By

```
Select P.ProductName, AVG(O.Count)  
As AvgOfCount from Product  
As P Inner Join [Order] As O  
On P.ProductId = O.ProductId  
Group by P.ProductId
```

کوئری‌های تو در تو:

اگر لازم باشد نتیجه یک کوئری در کوئری دیگر استفاده شود باید از عملگرهای زیر استفاده شود:

In, Exists, Not In, Not Exists, Distinct, Any, Some, All, Having

۱ - In, Exist

وجود یک عنصر را در یک مجموعه بررسی می‌کند.

In زمانی استفاده می‌شود که Sub Query و اطلاعات کمی برگرداند و Exists زمانی

استفاده می‌شود که ساب کوئری اطلاعات زیادی بازگرداند.

کوئری: لیست محصولات سفید رنگ را بدست آورید:

۱- یافتن محصولاتی که رنگشان سفید است

۲- یافتن اطلاعات محصولات سفید رنگ

Select * From Product
Where ProductId In (Select ProductId
From Product
Where color = "white")

کوئری: محصولاتی که قیمت آنها بین ۲۰۰ تا ۵۰۰ است:

Select * From Product
Where ProductId In (Select ProductId From Product
Where Price Between 200 and 500)

۲ - Not In, Not Exists

عدم وجود عنصری در مجموعه، تفاوت این دو همانند تفاوت In, Exists است.

کوئری: مشخصات تمام محصولات به جز آنهایی که رنگشان سفید است:

Select * From Product
Where ProductId Not In (Select ProductId
From Product
Where Color = "white")

کوئری: مشخصات محصولاتی که از آنها سفارشی ثبت نشده است:

Select * From Product
Where ProductId Not In (Select P.Product
From Product As P
Inner Join [Order] As O on
P.productId = O.ProductId)

۳ - Distinct

در جواب کوئری اطلاعات تکراری وجود نداشته باشد.
کوئری: لیست کاربرانی که برای آنها سفارشی ثبت شده:

```
Select Distinct C.CustomerId, FName, LName  
From Customer As C Inner Join [Order] As O On  
C.CustomerId = O.CustomerId
```

۴ - Some, Any

$A = \{1, 10, 15, 60\}$

$X > \text{Any } A$ یعنی x حداقل از یکی از اعضای A بزرگتر باشد

کوئری: محصولات که قیمت آنها از یکی از محصولات سفارش داده شده بیش‌تر باشد.

```
Select * From Product  
Where ProductId = Any (Select Product  
From [Order]  
Where Count > 3)
```

کوئری: محصولات که قیمت آنها از یکی از محصولات سفارش داده شده بیش‌تر باشد.

```
Select * From Product  
Where Price > Any (Select P.Price From Product  
A P Inner Join [Order]  
As O on P.ProductId = O.ProductId)
```

۵ - All

$x < \text{All } (A)$ یعنی x هایی که یکی از تمام مقادیر A کوچکتر باشد.

کوئری: لیست محصولات که از محصولات سفید رنگ ارزان‌تر باشد:

```
Select * From Product  
Where Price < All (Select Price From Product  
Where Color = "white")
```


۶- Having

بعد از دستور Where نمی‌توان از توابع استفاده کرد یعنی نمی‌توان نوشت:

```
Where Sum(O.Count) > 10
```

در این مواقع به جای Where از دستور Having استفاده می‌کنیم.

کوئری: لیست محصولات که مجموع سفارش آنها بیش از ۱۰ است:

```
Select P.ProductName, Sum (O.Count)
As orderCount From [Order]
As O
Inner Join Product As P On
P.ProductId = O.ProductId
GroupBy P.ProductName
Having Sum(O.Count) > 10
```

کوئری: نام مشتریانی که بیش از ۵ محصول سفارش داده‌اند:

```
Select C.FName, C.LName,
Sum (Count) As orderSum
From [Order] As O Inner Join
Customer As C On
O.CustomerId = C.CustmerId
Group By C.FName, C.LName
Having Sum (Count) > 5
```

رشته‌ها داخل *'Single Quotations'* قرار می‌گیرند



نه *"Double Quotations"* توجه شود!

جلسه نهم

:View

یک کوئری ذخیره شده است که برای سهولت در استخراج اطلاعات پیچیده از آن استفاده می‌شود.

یکی از راه‌های مناسب تقسیم یک گزارش پیچیده به بخش‌های کوچک‌تر استفاده از View است.

View تنها برای استخراج اطلاعات استفاده می‌شود.

Create View <Name> Select

مثال:

گزارش زیر را تهیه کنید:

- ۱- لیست مشتریان
- ۲- لیست محصولات
- ۳- لیست سفارشات
- ۴- لیست محصولاتی که بیش‌ترین فروش را داشته‌اند
- ۵- مشخصات مشتری که بیش‌ترین خرید را داشته

۱- لیست مشتریان:

```
Create view CustomerList As  
Select CustomerId, FName, LName, Cellphone  
From customer
```

بعد از دستور فوق اگر در پایین صفحه در کادر Message پیام زیر ظاهر شود یعنی

View با موفقیت ظاهر شده:

"Commands completed successfully"

* برای دیدن لیست View های تولید شده:

DB -> + View

* نحوه استفاده از view مانند استخراج اطلاعات از جدول است:

Select * From < View Name >

۲- تمرین: نام محصولات را بدست آورید.

۳- لیست سفارشات:

Create View OrderList As

Select C.FName, C.LName, P.ProductName,

P.Price, O.Count, P.Price * O.Count As Total

From productList As P Inner Join

[Order] As O On P.ProductId = O.ProductId Inner Join

CustomerList As C On

O.CustomerId = C.CustomerId

* در این مثال به جای Join کردن جداول، View های آنها را که اطلاعات کمتری دارند Join کرده ایم.

۴- لیست محصولات پرفروش:

```
Create View SumOfOrders As
Select P.ProductId, P.ProductName, Sum(Count)
As SumOfOrder
From dbo.[Order] As O Inner Join
ProductList As P On
P.ProductId = O.ProductId
Group By P.ProductId, P.ProductName
```

.....

```
Select Max (SumOfOrder) As
MaxOfOrder From SumOfOrders
```

.....

```
Select * From MaxOfOrders
```

۵- مشتری با بیشترین خرید:

عملیات مشابه مثال قبل است تنها میزان سفارش براساس کد مشتری است.

ساخت view برای محاسبه میزان سفارش هرکاربر:

```
Create View CustomerOrders As
Select C.CustomerId, FName, LName, Sum(O.Count)
SumOfOrder From CustomerList As
C Inner Join [Order] As O On C.CustomerId = O.CustomerId
Group By C.CustomerId, C.FName, C.LName
```

* Group By روی سه فیلد است زیرا CustomerList سه فیلد استخراج شده.

Create View CustomerMaxOrder As

Select * From Customerorders

Where SumOfOrder =

(Select Max (SumOfOrder) From
CustomerOrders)

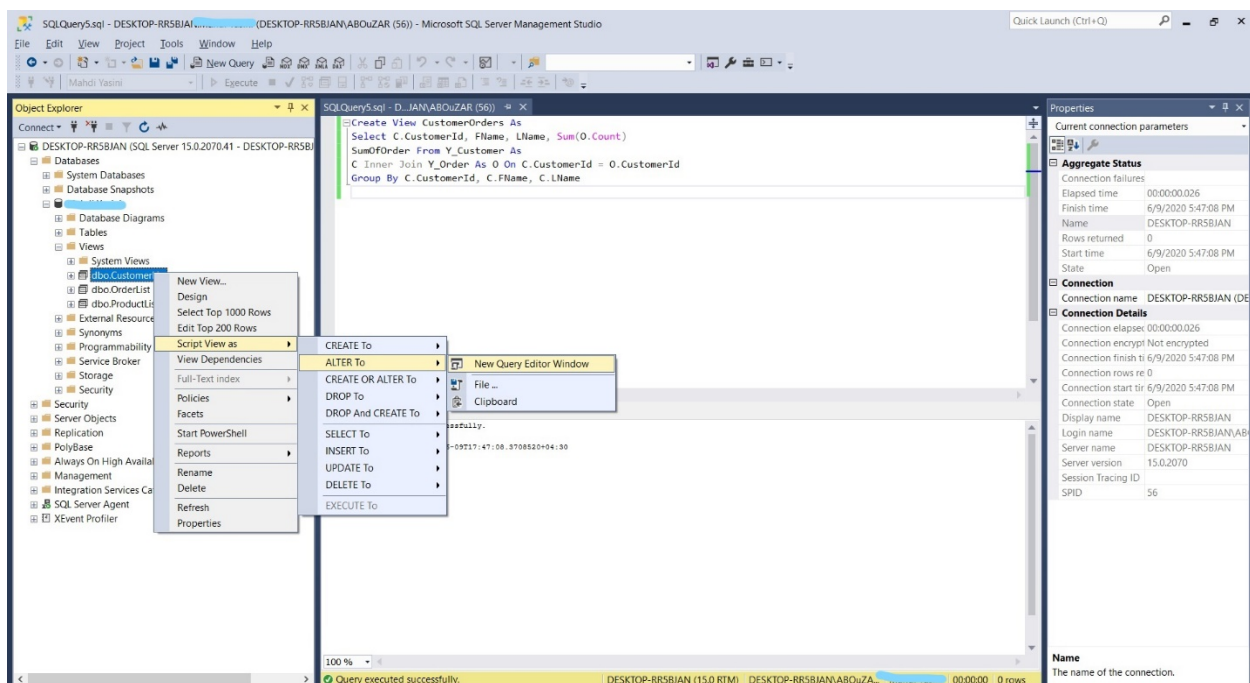
.....

Select * From CustomerMaxOrder

ویرایش View

Right Click on View -> Script view as -> Alter to -> New Query Editor

Window -> اعمال تغییرات -> Execute



محدودیت‌های View:

۱- تنها برای استخراج اطلاعات استفاده می‌شود

۲- نمی‌تواند پارامتر داشته باشد.

SP: stored Procedure

مثال:

بر اساس نام محصول میزان فروش هریک را نمایش دهید.

باید به گونه‌ای عمل کرد که کوئری بعد از ساهت بارها قابل استفاده باشد و با هربار تغییر نام محصول مجبور نباشیم کوئری را عوض کنیم.

قابلیت‌های SP

۱- دارای عملیات Insert و Update، Delete

۲- می‌تواند پارامتر داشته باشد.

مثال:

Create Procedure <SP Name>

پارامترها@

As


Begin

دستورات SQL

END

SPای بنویسید که مشخصات یک محصول را با استفاده از نام محصول جستجو کرده و نمایش دهد.

```
Create Procedure ProductSearch  
@name nvarchar(20)  
  
As  
  
Begin  
  
Select * From Product  
  
Where Productname = @name  
  
End
```


برای اجرا F5 یا Execute را فشار دهید. 
ظاهر شدن پیام **Commands completed successfully** به معنای آن است که ساخت SP با موفقیت انجام شده است.

اجرا:

```
Exec CustomerSearch  
@fname = 'Ali'  
@lname = 'Razavi'
```

مثال:

خریده‌های هرکاربر را با استفاده از نام خانوادگی او نمایش دهید.

برای مشاهده سفارش‌های مشتریان باید سه جدول Order، Customer و Product  با هم Join شوند این عملیات قبلاً در قالب دید OrderList ذخیره شده است در نتیجه می‌توان از آن استفاده نمود.

```
Create Procedure OrderSearch  
@lname nvarchar(30)
```

```
As
```

```
Begin
```

```
Select * From OrderList
```

```
Where LName = @Lname
```

```
End
```

اجرا:

```
Exec OrderSearch
```

```
@lname = 'Mohammadi'
```

مثال:

spای برای ثبت مشخصات محصول بنویسید.


```
Create Procedure InsertProduct
,@pname nvarchar (20)
,@color nvarchar (20)
@price int
As
Begin
Insert Into Product
(ProductName, Color, price)
Values (@pname, @color, @price)
End
```

اجرا:

```
Exec InsertProduct
@pname = 'Galaxy A5',
@color = 'white',
@price = 500
```



اگر در کادر Message پیام 1 row affected ظاهر شود یعنی ثبت اطلاعات با موفقیت انجام شده است.

جلسه دهم

پارامترهای اختیاری در SP:

اشکال SP آن است که اگر هنگام فراخوانی آن پارامتری داده نشود خطا گرفته می شود.

مثال: SP ای بنویسید که لیست سفارشات را بر اساس نام خانوادگی مشتری یا نام محصول نمایش دهد.

Create Procedure OrderSearchByParam

@Lname navchar (30) = Null;

@Pname navchar (20) = Null;

As

Begin

Select * From OrderList

Where LName = coalesce (@Lname, LName)

AND

ProductName = coalesce (@Pname, ProductName)

End

پارامترها مقدار اولیه را Null گرفته‌اند تا اگر مقدار دهی نشدند برای آنها Null در نظر گرفته شود

تابع coalesce با دو پارامتر تعریف شده این تابع اولین پارامتر غیر Null را برمی‌گرداند
یعنی اگر SP با @Lname برابر با "حسین" فراخوانی شود شرط به شکل زیر خواهد بود:
Where LName = N 'حسین'

و اگر در SP برای @Lname مقداری منظور نشود به شکر زیر خواهد بود:

Where LName = LName

اگر در قسمت شرط نام فیلد برابر با خودش شود، "شرط خنثی" ایجاد می‌شود انگار که اصلا شرطی تعریف نشده است. بنابراین SP فوق به ۴ شکل زیر قابل استفاده است.


Exec OrderSearchByParam (۱

Exec OrderSearchBy Param @lname = "Hosseini" (۲

Exec OrderSearchBy Param @Pname = "Iphone 8" (۳

Exec OrderSearchBy Param @Lname = "Hosseini", (۴

@Pname = "Iphone 8"

جستجوی پیشرفته یا ترکیبی به این روش قابل پیاده سازی است. 

ویرایش SP:

DB -> Programmability -> Stored Procedure -> click right on SP ->

Modify -> Change Whatever you want -> Execute

فصل ۶، محافظت از پایگاه داده

امکانات SQL برای محافظت عبارتست از:

Backup & Restore

Detach & Attach

DTS

ExportData

: Backup


تهیه نسخه پشتیبان باید در بازه‌ی مناسب انجام شود تا در صورت بروز مشکل بتوان از آن استفاده نمود. بهتر است این نسخه در حافظه‌ای مجزا ذخیره شود.

ص ۱۳۲:

Right Click On DB -> Tasks -> Backup

Backup set will expire (تعیین زمان انقضای نسخه)

Destination (محل ذخیره پشتیبان)

محل ذخیره پیش فرض همان محل نصب SQL است. برای تغییر این محل روی دکمه  Remove کلیک کنید تا محل پیش فرض حذف شود بعد بر روی دکمه Add کلیک کنید تا مکان جدید را تعیین کنید. در پنجره باز شده روی [...] کلیک کنید بعد از تعیین محل مقابل File Name نام فایل پشتیبان را تعیین کرده و برای آن پسوند bak. را انتخاب کنید و در آخر روی Ok کلیک کنید.

:Restore

برای ذخیره مجدد نسخه پشتیبان ابتدا باید DB ای که می‌خواهیم فایل پشتیبان آنرا روی سیستم منتقل کنیم حذف کنیم.

Right Click on DB -> Restore DB -> source: Device -> Click on [...] -> Add (set up backup directory) -> Ok

:Detach

غیرفعال کردن موقتی یک پایگاه داده

Right Click On DB -> Task -> Detach -> you have two choice:


Drop Connections (قطع ارتباط با پایگاه داده)

Update statistics (به هنگام سازی اطلاعات مربوط به امار بهینه سازی)

-> Ok

:Attach

فعال کردن پایگاه داده‌ای که Detach شده.

 باید فایل‌های mdf و ldf هر دو در یک مسیر قرار داشته باشد.

Click Right on DB -> Attach -> Add -> choose mdf file -> Ldf file will automatically select

:DTS: Data Transformation Services

۱- تبدیل اطلاعات موجود جداول DB به فرمت مورد نظر.

۲- وارد کردن اطلاعات سایر منابع اطلاعاتی به DB

:Import

می‌خواهیم اطلاعات یک فایل Excel را به پایگاه داده منتقل کنیم.

Create DB named DSTExample -> Right Click on DB -> Tasks ->
ImportData ->

Welcome to SQL Server Import & Export wizard -> Next -> choose &
data Source -> Data Source: Microsoft Excel

Excel file path: Browse -> Choose Directory File ->

First row has column names:

سطر اول فایل به عنوان نام ستونهای جدول تعیین می‌شود.

-> Next -> Choose & Destination

Destination: SQL Server Native Client

تعیین نام سرورای که SQL روی آن نصب شده (Server name)

انتخاب پایگاه داده‌ای که اخیرا ایجاد کرده اید (Data base)

->Next -> Specify Table Copy or Query

->Copy Data from one or more tables or views

انتخاب این گزینه باعث کپی شدن اطلاعات فایل اکسل به پایگاه داده می‌شود

->Next -> Selected Source Tables & views

فهرست Sheet های فایل نمایش داده می‌شود. هرکدام که انتخاب شود، اطلاعات آن به جدول تبدیل می‌شود.

->Next -> Save & Run Package

Run Immediately -> Finish

حال اگر DST Example را باز کنید خواهید دید که جدولی هم نام با Sheet در اکسل ایجاد شده.

:Export Data

می توان اطلاعات DB را به فرمت های Text, Access, Excel و ... تبدیل کرد.

Right click on DB -> Tasks -> Export Data ->

Welcome to SQL server Import & Export wizard -> Next ->

Choose a Data source

Data Source: SQL Server Native Client

Server name:

انتخاب نام سروری که SQL روی آن نصب شده

Database:

انتخاب پایگاه داده

->Next -> Copy data from one or more tables or views ->

-> Next -> Select Source Tables & views ->

مشاهده لیستی از جداول و viewها

انتخاب جدول مورد نظر ->

->Next -> Run Immediately

-> Next -> The execution was successful

-> Close

جلسه یازدهم

فصل ۷، امنیت پایگاه داده.

در windows server سرویسی وجود دارد به نام Active Directory برای مدیریت کاربران، تعیین سطح دسترسی، تعریف گروه‌های کاربری و اعمال محدودیت دسترسی این امکانات در SQL نیز وجود.

ایجاد مدیریت کاربران و گروه‌های کاربری:

Object Explorer -> Security

نقش‌های پیش‌فرض SQL:

این نقش‌ها در بخش Server Roles قرار دارند برای نقش مجموعه محدودیت‌ها، قوانین و مقررات خاصی مشخص شده است.

:Bulk admin

کار با حجم بالای داده نیاز به عضویت در این گروه دارد. مثل وارد کردن اطلاعات بسیار حجیم به پایگاه داده.

:DB Creator

اعضا این گروه اجازه ایجاد پایگاه داده را دارند ولی اجازه مدیریت آن را ندارند.

:Disk admin

مدیریت عملیات مرتبط با حافظه مثل تهیه فایل پشتیبان (Back up)

:Public

تمام کاربران به طور پیش‌فرض عضو این گروه‌اند. محدودیتی برای آن‌ها در نظر گرفته نشده است.

:Process admin

توانایی اعطا یا سلب دسترسی افراد به پایگاه داده.

:Security admin

مدیریت کاربران login شده، فعال یا غیر فعال سازی کاربران اجازه دسترسی به پایگاه داده.

:Server admin

کلیدی ترین نقش، توانایی اعمال تنظیمات پایگاه داده حتی خاموش کردن سرویس های پایگاه داده.

:Sys admin

بالاترین سطح دسترسی

:Setup admin

مدیریت سرورهای متصل به هم.

تعریف حساب کاربری جدید:

Object Explorer -> Security -> Right click login ->

(2 option), 1) SQL Server Authentication (Member with password)

2) Windows Authentication (Member without password)

Enforce password policy:

رمز ساده و قابل حدس نمی توان انتخاب کرد.

Enforce password expiration:

تعیین طول عمر رمز عبور

User must change password on next login:

کاربر پس از اولین عبور باید رمز عبور را تغییر دهد

Default DB:

انتخاب نام پایگاه داده پیش فرض که کاربر هنگام ورود می تواند از آن استفاده کند.

Default Language:

انتخاب زبان پیش فرض پس از اعمال تنظیمات فوق روی گزینه

User Mapping

کلیک کنید

اجرای -> Close DB -> Ok -> تعیین نقش کاربر در قسمت DB Role -> choose DB ->

مجدد

فصل ۸، اتصال برنامه به پایگاه داده

در این فصل نحوه ارتباط برنامه تحت ویندوز با پایگاه داده و انجام اعمال ذخیره سازی، ویرایش، حذف و گزارش گیری از اطلاعات آموزش داده می شود.

Visual Studio 2017 -> File -> New -> Project -> Visual C#: Windows

classic Desktop -> Windows Form App (.Net Framework) -> تعیین نوع پروژه

-> Name: نام پروژه

Location: تعیین محل ذخیره پروژه

->Ok -> Solution Explorer (حاوی محتوای پروژه) , Properties (تنظیم مشخصات)

(حاوی کنترل های لازم برای ساخت پروژه) Toolbox, (کنترل انتخاب شده

اگر قسمت های فوق را مشاهده نمی کنید از منوی view گزینه مورد نظر را انتخاب کنید.

روش های برقراری ارتباط بین برنامه و پایگاه داده:

۱- ADO, NET: تکنولوژی قدیمی و کم استفاده

۲- EF (Entity Framework): تکنولوژی فعلی رایج

مدل سازی:

یعنی تبدیل جدول به class و فیلد به property

مدل سازی با EF:

انتخاب گزینه Data -> New Item -> Add -> Right click on DB name

Add -> Name (Project Name) -> انتخاب گزینه Ado .NET Entity Data Model

Server -> New Connection -> Next -> EF Designer from DB -> Add -> (نام سروری که اس کیو ال روی آن نصب شده است) Name

Select or enter a DB Name:

انتخاب پایگاه داده ای که می خواهید به آن وصل شوید

-> Test connection succeeded -> Ok -> connection properties: Ok ->

save connection setting in App. Config as: -> نمایش مشخصات ارتباط ساخته شده

-> set EF version -> EF 6.x -> Next -> اینجا نام پروژه نوشته شده

Finish -> انتخاب جداولی که قصد دارید آن ها را مدل سازی کنید Tables:dbo:

بستن پنجره -> ctrl + s -> مدل سازی انجام شد

فایلی با نام پروژه و پسوند .edmx ایجاد شده که حاوی اطلاعات و کلاسهای مدل سازی شده است. (ص ۱۶۶)

جلسه ۱۲، ساخت پروژه (جلسه آخر)

هنگام ساخت پروژه تحت ویندوز، فرمی با نام Form1 ایجاد می شود رو نام آن کلیک راست کرده و آن را Delete می کنید.

MDI Form:

با این فرم می توان اطلاعات مختلف برنامه را مدیریت کرد.

افزودن فرم MDI به پروژه:

Solution Explorer -> Right click on project name -> Add: New Item ->

MDI Parent Form -> Name: **firm Main** -> Ok

حذف کنترل‌های پیش فرض MDI:

قصد داریم از کنترل‌های دلخواه استفاده کنیم بنابراین باید کنترل‌های پیش فرض MDI را حذف کنیم. گزینه‌های Status Strip، Tool Strip، Menu Strip و Tool Tip را انتخاب کرده و حذف می‌کنیم

حذف کدهای کنترل‌های حذف شده:

حذف تمام متدها **جز** firmMain () -> ورود به محیط کد نویسی -> F7

تعیین فرم اولیه برنامه جهت اجرا:

Solution Explorer -> Double click on Program.cs -> Application.Run(new firmMain ()) اصلاح متد:

ثبت اطلاعات:

برای هر یک از عملیات پایگاه داده باید یک فرم طراحی کرد

Solution Explorer -> Right click on project name -> Add: New Item -> windows Form -> Name: firmAddCustomer -> Add

تغییر خواص فرم:

F4 -> Properties -> Right To Left: yes, Text: ثبت مشخصات کاربر

متناسب با فیلدهای جدول Customer به آن Button، Label و Text Box اضافه می‌کنیم:

ثبت مستحقات کاربر

نام

نام خانوادگی

تلفن همراه

آدرس

ثبت اطلاعات

GANJINEH

خاصیت Text مربوط به Label ها را به شکل زیر تعیین کنید:

- نام
- نام خانوادگی
- تلفن همراه
- آدرس

خاصیت Name مربوط به Text Box ها را به شکل زیر تعیین کنید:

- txtFirstName
- txtLastName
- txtPhone
- txtAddress

خاصیت Text مربوط به Button را برابر "ثبت اطلاعات" قرار دهید.

خاصیت Name مربوط به Button را برابر btnSave قرار دهید.

روی دکمه ثبت اطلاعات کلیک کرده و وارد محیط کد نویسی شوید.

یک شی از کلاس اصلی به نام (نام پروژه) ایجاد کرده و نام آن را db بگذارید.

```
MyProjectEntities db = new MyProjectEntities;
```

دستورات زیر را در رویداد btn Save-Click وارد کنید:

```
Private void btnSave-click (object sender, EventArgs e) {  
    customer customer = new Customer();  
    customer.FirstName = txtFirstName.Text;  
    customer.LastName = txtLastName.Text;  
    customer.Phone = txtPhone.Text;  
    customer.Address = txtAddress.Text;  
    db.Customers.Add(customer); // شی به پایگاه داده اضافه می‌شود  
    db.SaveChanges(); // ذخیره اطلاعات  
    MessageBox.Show("اطلاعات با موفقیت ثبت شد", MessageBoxButtons.Ok);  
}
```

فشردن کلید F5 باعث اجرای برنامه و نمایش فرم اصلی می‌شود.

ایجاد منو برای فرم اصلی

باید در این فرم منویی طراحی کنیم که با گزینه‌های آن بتوان به سایر فرم‌ها دسترسی داشت

Solution Explorer -> Double click on firmMain -> Menu & Toolbar ->

ToolBox -> Double click on Menu Strip -> به فرم اضافه شد -> انتخاب منو ->

Properties: Right To Left: True

Type Here: مشتری

ایجاد زیر منو:

بعد از افزودن یک منو می‌توان به آن زیر منو یا منوی هم سطح اضافه کرد.

زیر منوهای زیر را ایجاد کنید:

- مشتری جدید
- لیست مشتریان

روی مشتری جدید دابل کلیک کرده و وارد محیط کد نویسی شوید در اینجا باید دستورات مربوط به اجرای فرم افزودن مشتری درج شود:

```
frmAddCustomer form = new frmAddCustomer();
```

```
form.MdiParent = this();
```

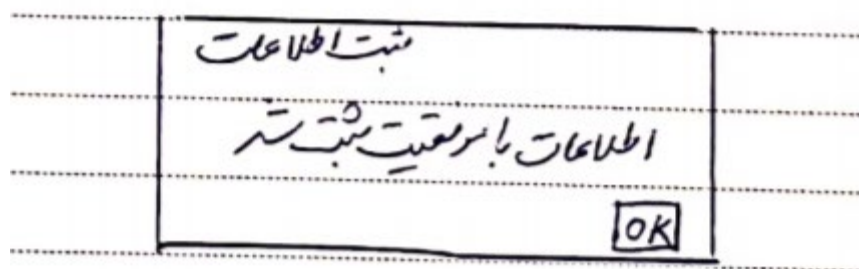
```
form.Show();
```

با فشردن کلید F5 و یا کلیک بر روی Start برنامه اجرا می‌شود.

وارد کردن اطلاعات:

می‌خواهیم اطلاعات مشتری را وارد کنیم. روی گزینه مشتری جدید کلیک می‌کنیم تا فرم مربوطه نمایش داده شود. اطلاعات یک مشتری را وارد کرده روی دکمه "ثبت اطلاعات" کلیک می‌کنیم.

اگر دستورات ارتباط با پایگاه داده به درستی نوشته شده باشد پیام زیر نمایش داده می‌شود:



بررسی درستی ثبت اطلاعات:

برای اطمینان از صحت ثبت اطلاعات جدول مشتری در پایگاه داده، کوئری زیر را ایجاد می‌کنیم.

```
Select * From Customer
```

اگر اطلاعات جدید نمایش داده شد یعنی عمل ثبت به درستی انجام شده است.

نمایش List:

می‌خواهیم امکان مشاهده List مشتریان را فراهم کنیم تا مجبور نباشیم از محیط SQL استفاده کنیم.

فرمی با نام frmCustomerList به پروژه اضافه کنید:

لیست مشتریان Text:

ToolBox -> Data -> Data Grid View (افزودن کنترل) -> Name: gridCustomer,

Right to left: yes -> دابل کلیک روی قسمت خالی فرم

```
MyProjectEntities db= new MyProjectEntities();
```

نوشتن دستورات استخراج و نمایش اطلاعات در رویداد Load:

```
Var data = db.Customers.ToList();
```

```
grdCustomer.DataSource = data;
```

در فرم اصلی برنامه روی منوی لیست مشتریان دابل کلیک کنید نوشتن دستورات فراخوانی لیست مشتریان:

```
frmCustomerList form=new;
```

```
frmCustomerList();
```

```
form.MdiParent = this;
```

```
form.Show();
```

برنامه را اجرا کنید از منوی مشتری گزینه لیست مشتریان را انتخاب کنید.

حذف اطلاعات:

میخواهیم برای فرم CustomerList گزینه حذف اطلاعات داشته باشیم، زیر GridView یک BUTTON اضافه می‌کنیم:

Name: btnDelete

Text: Delete

روی دکمه Delete دابل کلیک کنید.

کدهای زیر را وارد کنید:

```
Int id =  
convert.ToInt32(grdCustomer.CurrentRow.Cells["CustomerId"].Value);  
Customer customer = db.Customer.Find(id);  
(مشتری با ای دی مشخص شده در شی ایجاد شده قرار گرفت )  
db.Customers.Remove(customer); // حذف مشتری  
db.SaveChanges(); // ذخیره تغییرات  
MessageBox.Show("حذف", "عملیات حذف با موفقیت انجام شد"); // نمایش پیام به کاربر
```

بروزرسانی محتوای GridView:

بعد از حذف اطلاعات باید که رویداد frmCustomerList-Load() تبدیل به یک متد شود:

```
Private void LoadData(){  
    Var data = db.Customer.ToList();  
    grdCustomer.DataSource = data;  
}
```



```
Private void frmCustomerList_Load (object sender, EventArgs e) {  
    LoadData();  
}
```

```
Private void btnDelete_click (object sender, EventArgs e) {  
    Int id = convert.ToInt32(grdCustomer,  
    CurrentRow.Cells["CustomerID"].value);  
    Customer customer = db.Customer.Find(id);  
    db.Customer.Remove(customer);  
    db.SaveChanges();  
    MessageBox.Show("حذف", "عملیات حذف با موفقیت انجام شد");  
    LoadData();  
}
```

حال برنامه را اجرا کرده و وارد صفحه لیست مشتریان شده یک ردیف را انتخاب کرده و روی دکمه Delete کلیک کنید.

پایان

پ.ن: دوست عزیزم در صورت یافتن هرگونه مشکل نگارشی، اصلاح بفرمایید با تشکر و احترام