In the name of god

SQL Server

DDL & DML

Amirkabir University of Technology

Behnaz Motavali
bs.motavali@yahoo.com

Database Languages

DDL

- Data Definition Language
- Used to define data **structures**
- CREATE, DROP, ALTER, RENAME, …
- Database, Table, View, Index, …

DML

- Data Manipulation Language
- Used to manipulate **data itself**
- SELECT, INSERT, UPDATE, DELETE, read-only queries, …

Data Definition Language (DDL)

The CREATE TABLE Statement

- CREATE TABLE *[table name]* (*[column definitions]*) *[table parameters]*

- CREATE TABLE table_name
  (
  column_name1 data_type ,
  column_name2 data_type ,
  column_name3 data_type ,
  ....
)
- Example:
  - CREATE TABLE Persons
    (
    P_Id int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
)

P_Id	LastName	FirstName	Address	City

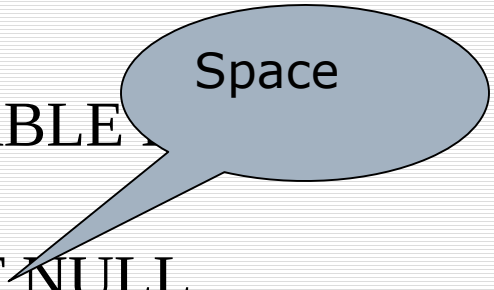
SQL Constraints

- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK
- DEFAULT

SQL NOT NULL Constraint

- The following SQL enforces the "P_Id" column and the "LastName" column to not accept NULL values:

- CREATE TABLE
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)



SQL UNIQUE Constraint

- The following SQL creates a UNIQUE constraint on the "P_Id“ and LastName column when the "Persons" table is created:

- CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CONSTRAINT uc_PersonID UNIQUE (P_Id,LastName)
)

P_Id int NOT NULL UNIQUE,

SQL PRIMARY KEY Constraint

- The following SQL creates a PRIMARY KEY on the "P_Id" column when the "Persons" table is created:

- CREATE TABLE Persons
  (
  P_Id int NOT NULL PRIMARY KEY,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255)
)

SQL PRIMARY KEY Constraint

- To allow naming of a PRIMARY KEY constraint, and for defining a PRIMARY KEY constraint on multiple columns, use the following SQL syntax:
- CREATE TABLE Persons
  (
  P_Id int NOT NULL,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255),
  CONSTRAINT pk_PersonID PRIMARY KEY (P_Id,LastName)
)

primary key(P_ID,LastName)

SQL FOREIGN KEY Constraint

برای رسیدن به قوانین جامعیت داده ها (قانون جامعیت ارجاعی) از کلید خارجی استفاده می شود ، که بر روی جدول فرزند پیاده سازی می شود . در هنگام تعریف کلید خارجی توجه به نکات زیر مهم است :

In Parent Table	Status	In Child Table
Select	Nothing	Nothing
Insert	Nothing	Check If exists in Parent
Update	No Action	Return Error
	Cascade	Update Foreign Key in Child Rows
	Set Null	Sets Null Value To Foreign Key if Nullable Or Return Error if Not Nullable
	Set Default	Set Default Value if Exists in Parent Table Or Return Error if Not Exists.
Delete	No Action	Return Error
	Cascade	Delete Child Rows
	Set Null	Sets Null Value To Foreign Key if Nullable Or Return Error if Not Nullable
	Set Default	Set Default Value if Exists in Parent Table Or Return Error if Not Exists.

هنگام Insert / Update در جدول فرزند :

در این هنگام مقادیر وارد شده برای ستون کلید خارجی در جدول فرزند ، با مقادیر موجود در کلید اصلی جدول پدر مقایسه شده ، در صورت عدم وجود این مقدار در جدول پدر، اجازه عملیات صادر نمی شود .

هنگام Delete / Update در جدول پدر :

در این هنگام امکان انتخاب یکی از حالتهای زیر در جدول فرزند وجود دارد :

1) No Action
2) Cascade
3) Set Null
4) Set Default

SQL FOREIGN KEY Constraint

#1:

```
CREATE TABLE Orders
(
O_Id int NOT NULL PRIMARY KEY,
OrderNo int NOT NULL,
P_Id int FOREIGN KEY REFERENCES Persons(P_Id)
)
```

#2:

```
CREATE TABLE Orders
(
O_Id int NOT NULL,
OrderNo int NOT NULL,
P_Id int,
PRIMARY KEY (O_Id),
CONSTRAINT fk_PerOrders FOREIGN KEY (P_Id) REFERENCES Persons(P_Id)
)
```

SQL CHECK Constraint

- The following SQL creates a CHECK constraint on the "P_Id" column when the "Persons" table is created. The CHECK constraint specifies that the column "P_Id" must only include integers greater than 0.
- CREATE TABLE Persons
  (
  P_Id int NOT NULL,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255),
  CONSTRAINT chk_Person CHECK (P_Id>0 AND City='Sandnes')
)

SQL DEFAULT Constraint

- The following SQL creates a DEFAULT constraint on the "City" column when the "Persons" table is created:

- CREATE TABLE Persons
  (
  P_Id int NOT NULL,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255) DEFAULT ‘Tehran'
)

مرور

دستور ایجاد پایگاه داده ها

- create database نام پایگاه ها
- create database sale

دستور ایجاد جدول

- CREATE TABLE نام جدول
  (
  نام ستون ۱ نوع ستون ۱ [not null] [unique] ,
  نام ستون ۲ نوع ستون ۲ [not null] [unique] ,
  نام ستون ۳ نوع ستون ۳ [not null] [unique] ,
  .... ,
  primary key(نام ویژگیهای تشکیل دهنده کلید اصلی),
  foreign key (نام ویژگیهای کلید خارجی ۱) references نام جدول مورد نظر,
  foreign key (نام ویژگیهای کلید خارجی ۲) references نام جدول مورد نظر,
  .....
  [check شرط مورد نظر]
)

زبان تعریف داده ها DDL

- معروفترین انواع داده در SQL

نوع داده ای	بازه
integer	اعداد صحیح
smallint	اعداد صحیح
decimal(p,q)	اعدادی با p رقم و q رقم اعشاری در سمت راست
float	اعداد اعشاری با ممیز شناور
date	تاریخ با فرمت yyyymmdd
time	زمان با فرمت hhmmss
char(n)	رشته کارکتری با طول ثابت n
varchar(n)	رشته کارکتری با طول متغیر کوچکتر یا مساوی n
nchar(n)	رشته کارکتری با طول ثابت n (پشتیبانی از زبان فارسی)
nvarchar(n)	رشته کارکتری با طول متغیر کوچکتر یا مساوی n (پشتیبانی از زبان فارسی)

جداول بکار رفته در اسلاید

- جدول اطلاعات تولید کنندگان
- S(s#,sname,city)
- جدول اطلاعات محصولات
- P(p#,pname,color)
- جدول اطلاعات پروژه ها
- J(j#,jname,city)
- جدول اطلاعات فروش
- SPJ(s#,p#,j#,qty)

زبان تعریف داده ها DDL

- مثال:
  - جدول S را ایجاد کنید بطوریکه کاربر مجاز نباشد هیچ تاپلی با sname خالی یا تکراری در این جدول درج کند

```
Create table S
(
s# char(2),
sname nchar(30) not null unique,
city nchar(20),
primary key(s#)
)
```

زبان تعریف داده ها DDL

- مثال:
  - جدول SPJ را ایجاد کنید بطوریکه بازه مجاز برای qty اعداد بین ۵۰۰ تا ۲۰۰۰۰ باشد.

```
Create table SPJ
(
s# char(2),
p# char(2),
j# char(2),
qty int,
primary key(s#,p#,j#),
foreign key(s#) references S,
foreign key(p#) references P,
foreign key(j#) references J,
check(qty>500 and qty<=20000)
)
```

The ALTER Statement

- The *ALTER* statement modifies an existing database object

- ALTER *objecttype objectname parameters*

  - ALTER TABLE table_name add (column_name data_type)
  - ALTER TABLE table_name modify (column_name data_type)

زبان تعریف داده ها DDL

- **اضافه** کردن یک ستون جدید به یک جدول
- alter table نام جدول add(نام ستون مشخصات ستون جدید)
- مثال
- در جدول s ستون جدیدی به نام tel برای درج شماره تلفن تولیدکنندگان اضافه کنید.
  - alter table S add(tel char(10))
- **تغییر** مشخصات یک ستون از یک جدول
- alter table نام جدول modify(نام ستون مشخصات جدید)
- مثال
- در جدول S طول ستون Sname را از ۳۰ کاراکتر به ۲۰ کاراکتر تغییر دهید
  - alter table S modify(sname nchar(20) not null unique)

The DROP Statement

- The *DROP* statement destroys an existing database, table, index, or view

- DROP *objecttype objectname*

  - DROP TABLE table_name
  - DROP DATABASE database_name

زبان تعریف داده ها DDL

- **حذف** یک جدول
- drop table نام جدول
- مثال
- جدول s را حذف کنید
  - drop table S

The RENAME Statement

- The *RENAME* statement is used to rename a database table

- RENAME TABLE old_name to new_name

Data Manipulation Language (DML)

The SELECT Statement

- The ***SELECT*** statement returns a result set of records from one or more tables
- is the most commonly used data query language(DQL) command
- *SELECT ... FROM ... WHERE ...*
- The SELECT statement has many optional clauses:
  - WHERE specifies which rows to retrieve
  - GROUP BY groups rows sharing a property so that an aggregate function can be applied to each group
  - HAVING selects among the groups defined by the GROUP BY
  - ORDER BY specifies an order in which to return the rows
  - AS provide an alias which can be used to temporarily rename table or columns

Sub Query

Sub Query حاوی یک دستور Select است که فقط یک مقدار را بر می گرداند به عبارت دیگر Result Set حاصل از یک Sub Query جدولی است تنها دارای یک سطر و یک ستون .

از نتیجه Sub Query می توان در دستورات Select و یا Where استفاده نمود .

مثال :

```
Use [Lab-Inventory]
Select
        (Select   Title        From   Item
          Where   Item.ID  =  Inventory.Item_ID)  As Item ,
        (Select   Title        From   Color
          Where   Color.ID     = Inventory.Item_ID)  As Color ,
        Quantity
  From
        Inventory
```

Nested Query

شامل یک یا چند دستور Select است که در دستور From قرار می گیرد

هنگامیکه بخواهیم مقادیری خاص از یک Result Set به عنوان محدوده بازیابی اطلاعات قرار گیرند ، از Nested Query استفاده می کنیم .

مثال :

```
Use [Lab-EasyShop]
Select
        Person.FirstName ,
        Person.LastName ,
        ShopOrder.Date
  From
        (Select  *     From   Customer
          Where     ID     Between   1 And 2        )  As Person,
        ShopOrder
  Where
        Person.ID  =   ShopOrder.Customer_ID
```

Initcap Function

- In oracle..

  select initcap('abcd') from name;

  returns

  Abcd

- In MS SQL

```
select upper(left(colName, 1)) + substring(colName, 2,
    len(colName)) as colName
```

توابع رشته ای در SQL Server

Ascii	مقدار کد Ascii مربوط به کاراکتر مورد نظر را برمی گرداند
Char	یک کد Ascii مربوط را به یک کاراکتر معادل تبدیل می کند
Unicode	مقدار کد Unicode مربوط به کاراکتر مورد نظر را برمی گرداند
NChar	یک کد Unicodeمربوط را یک کاراکتر معادل تبدیل می کند
CharIndex	در یک رشته محل شروع یک عبارت را اعلام می کند
Left	تعداد مشخص کاراکتر از سمت چپ رشته را بر می گرداند
Right	تعداد مشخص کاراکتر از سمت راست رشته را بر می گرداند
Substring	تعداد مشخص کاراکتر از محل مشخص شده در رشته را بر می گرداند
Lower	رشته را به حروف کوچک تبدیل می کند
Upper	رشته را به حروف بزرگ تبدیل می کند
Replace	در کل رشته مورد نظر عبارت خاصی را با عبارت دیگری جایگزین می کند
LTrim	کل فضا های خالی سمت چپ یک رشته را حذف می کند
RTrim	کل فضا های خالی سمت راست یک رشته را حذف می کند
Space	به تعداد مورد نظر ، فضای خالی بر می گرداند
Reverse	یک رشته را بصورت برعکس بر می گرداند
Len	طول یک رشته را بر می گرداند.
Str	یک مقدار عددی را به یک رشته تبدیل می کند.
Replicate	یک رشته را به تعداد مشخص تکرار می کند .

انواع Join ها

یکی دیگر از روشهای ترکیب و تلفیق اطلاعات در جداول مختلف ، استفاده از Join هاست .

سه نوع Join وجود دارد :

۱- Inner Join

۲- Outer Join

۳- Cross Join

تلفیق اطلاعات با استفاده از Join ها علاوه بر پوشش دادن طیف وسیع تری از نحوه تلفیق اطلاعات ، از حجیم شدن کد در دستور Where و همچنین کند شدن بر اثر فیلتر کردن اطلاعات جلوگیری می کند.

Join ها در دستور From بکار می روند.

Inner Join

برای تلفیق اطلاعات دو یا چند جدول که در رابطه منطقی دارای مقادیر متناظر هستند ، بکار می رود .

قالب کلی آن به شکل زیر است :

جدول ۱

Inner Join

جدول ۲

On **عبارت شرطی برای ایجاد یک رابطه منطقی**

مثال :

```
Select
        Item.Title ,
        Inventory.Quantity
  From
        Item
        Inner Join
        Inventory
          on    Item.ID    =   Inventory.Item_ID
```

Inner Join

مثال :

```
Select
        Item.Title              As Item,
        Color.Title             As Color,
        Inventory.Quantity
  From
        Item
          Inner Join
        Inventory
          on    Item.ID     =   Inventory.Item_ID
          Inner Join
        Color
          on    Color.ID    =   Inventory.Color_ID
```

Outer Join

برای تلفیق اطلاعات دو یا چند جدول که در رابطه منطقی حتی دارای مقادیر متناظر نیستند ، بکار می رود .

به سه صورت می تواند وجود داشته باشد :

۱- Left Outer Join

۲- Right Outer Join

۳-Full Outer Join

قالب کلی آن به شکل زیر است :

جدول ۱

{Left | Right | Full} Outer Join

جدول ۲

On

عبارت شرطی برای ایجاد یک رابطه منطقی

Outer Join

مثال :

```
Select
        Item.Title ,
        Inventory.Quantity
   From
        Item
           Left Outer Join
        Inventory
           on   Item.ID   =   Inventory.Item_ID
```

Outer Join

مثال :

```
Select
        Item.Title ,
        Inventory.Quantity
  From
        Item
          Left Outer Join
        Inventory
          on        Item.ID   =   Inventory.Item_ID
```

مثال بالا بدون خروجی Null با استفاده از تابع IsNull به صورت زیر در می آید :

```
Select
        Item.Title ,
        IsNull(Inventory.Quantity,0)
  From
        Item
          Left Outer Join
        Inventory
          on        Item.ID   =   Inventory.Item_ID
```

Cross Join

خروجی Cross Join ، حاصلضرب دکارتی مجموعه رکوردهای دو جدول می باشد .

دقت کنید : در Cross Join قسمت On و عبارت شرطی مرتبط کننده جداول حذف می شود

مثال :

```
Select
        Item.Title          As Item,
        Color.Title         As Color
  From
        Item
           Cross Join
        Color
```

The INSERT Statement

- The *INSERT* statement adds one or more records to any single table

- INSERT INTO table_name (column_name1[,column_name2,…]) VALUES (value1[, value2, …])

- number of columns and values must be the same
- If a column is not specified, the default value for the column is used
- The values specified (or implied) must satisfy all the applicable constraints

The UPDATE Statement

- The *UPDATE* statement changes the data of one or more records in a table

- UPDATE table_name SET column1= value1[,column2= value2] [WHERE condition]

- Either all the rows can be updated, or a subset may be chosen using a condition
- The updated value must not conflict with all the applicable constraints
- The user must have data manipulation privileges

The DELETE Statement

- The *DELET* statement removes one or more records from a table

- DELETE from table_name [WHERE condition]

- Any rows that match the condition will be removed from the table
- If the WHERE clause is omitted, all rows in the table are removed
- can cause triggers to run that can cause deletes in other tables