EIE 3112
Structured Query Language (SQL)
Part 1

T. Connolly and C. Begg, "*Database Systems: A Practical Approach to Design, Implementation, and Management*," 6th Edition, Chapter 6&7, Pearson, 2015. (5th Edition is also fine)

You Will Learn

- What is SQL
- Data Types in SQL
- Creating/Deleting Databases and Tables
- Constraints on Foreign Keys
- Retrieve Data from Database
- Data manipulation

What is SQL

- Structured Query Language
  - International standard
  - Database definition: create new DB and tables; modify table definition
    - `CREATE TABLE`, `ALTER TABLE`
  - Manipulation: retrieval and modification of rows
    - `Select, insert, update, delete`
  - Control: integrity and security constraints
    - `Grant, revoke`

Data Types in SQL

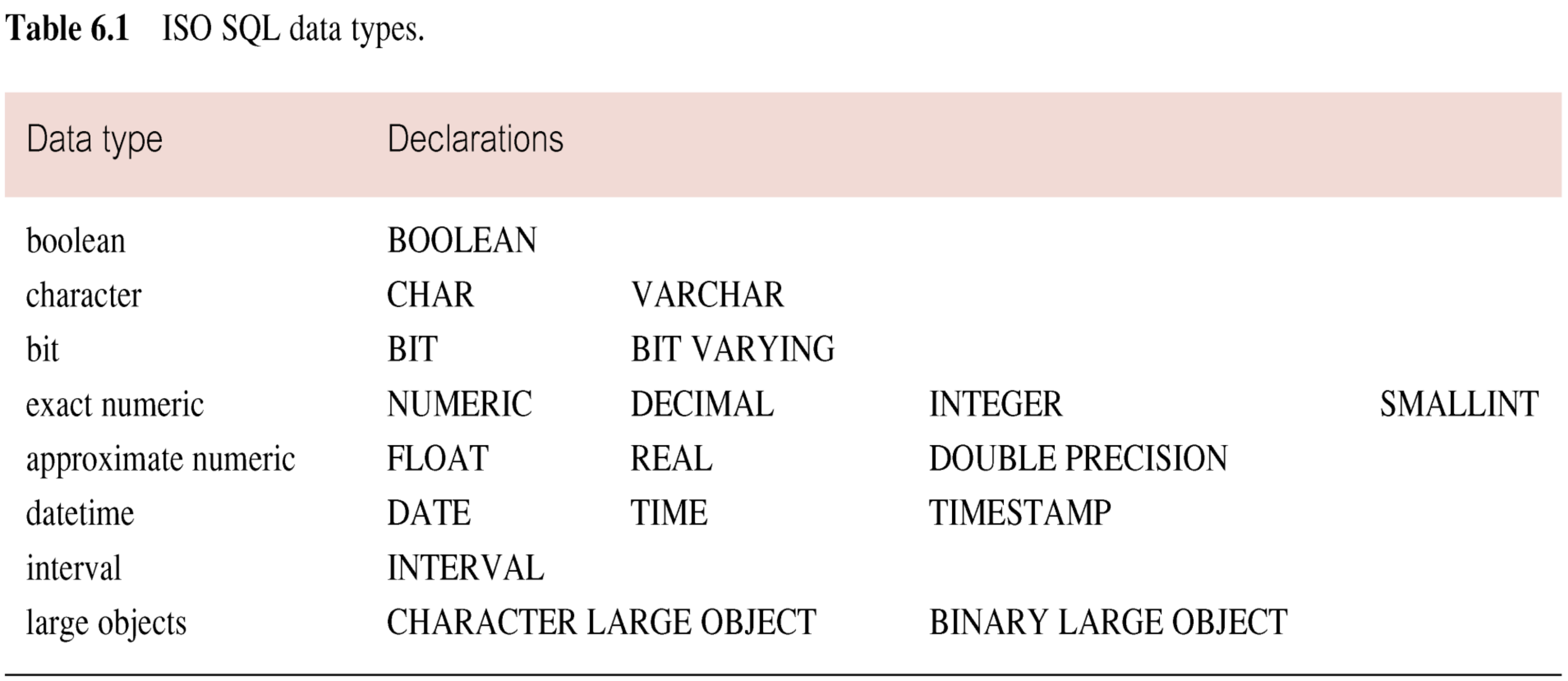
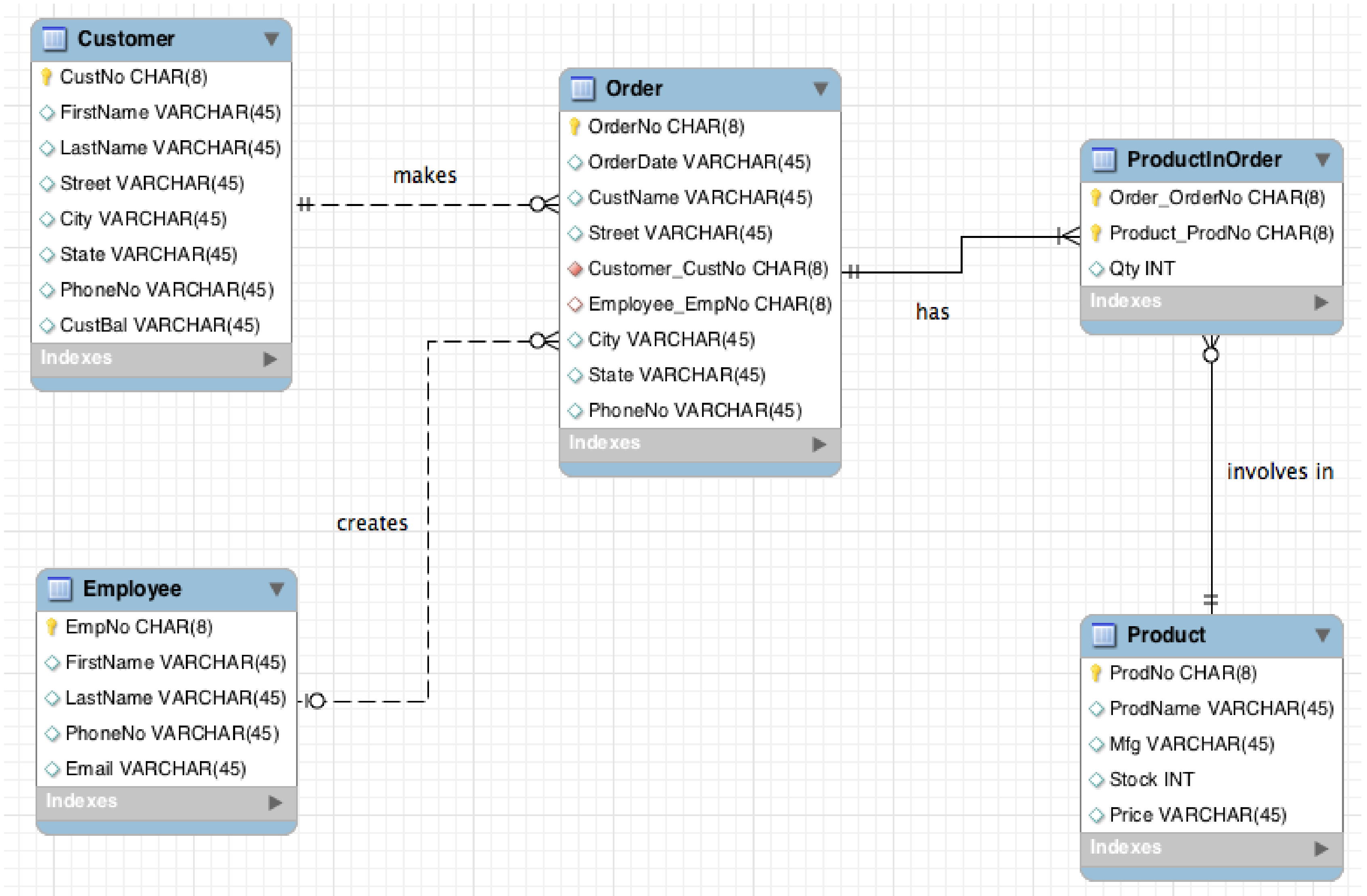


Table 6.1 ISO SQL data types.

Data type	Declarations			
boolean	BOOLEAN			
character	CHAR	VARCHAR		
bit	BIT	BIT VARYING		
exact numeric	NUMERIC	DECIMAL	INTEGER	SMALLINT
approximate numeric	FLOAT	REAL	DOUBLE PRECISION	
datetime	DATE	TIME	TIMESTAMP	
interval	INTERVAL			
large objects	CHARACTER LARGE OBJECT		BINARY LARGE OBJECT	

SQL by Example

- We will use the following example to learn SQL



Creating Databases

- When creating a database, it is important to understand the differences between database schemas and database instances
- Database Schemas
  - A database schema is the definition that describes the entire configuration of the database, including all of its tables, relations, index, etc.
  - Database schema is specified during database design
- Database Instance
  - The data in the database at any particular point in time is called a database instance.

Creating/Deleting Databases

Syntax:

```
CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name
    [create_specification] ...

create_specification:
    [DEFAULT] CHARACTER SET [=] charset_name
  | [DEFAULT] COLLATE [=] collation_name
```

```
DROP {DATABASE | SCHEMA} [IF EXISTS] db_name
```

Example (from Lab1):

Delete database `Lab1` if it exists

```
DROP DATABASE IF EXISTS Lab1;
CREATE DATABASE IF NOT EXISTS Lab1;
USE Lab1;
```

Creating/Deleting Tables

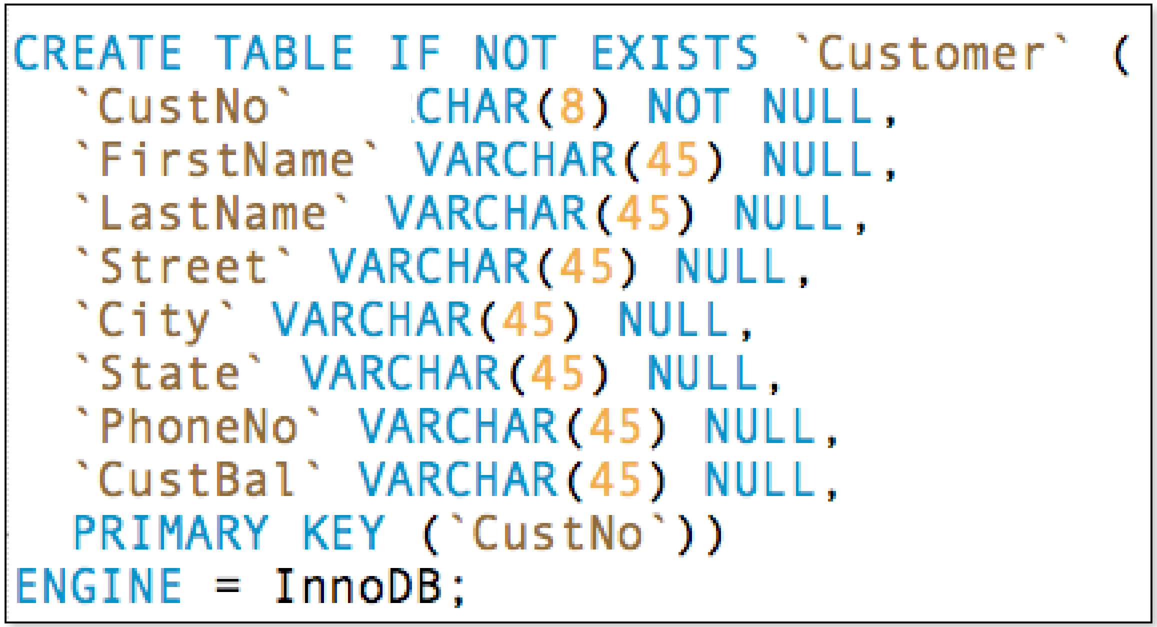
Syntax:

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    (create_definition,...)
    [table_options]
    [partition_options]
```

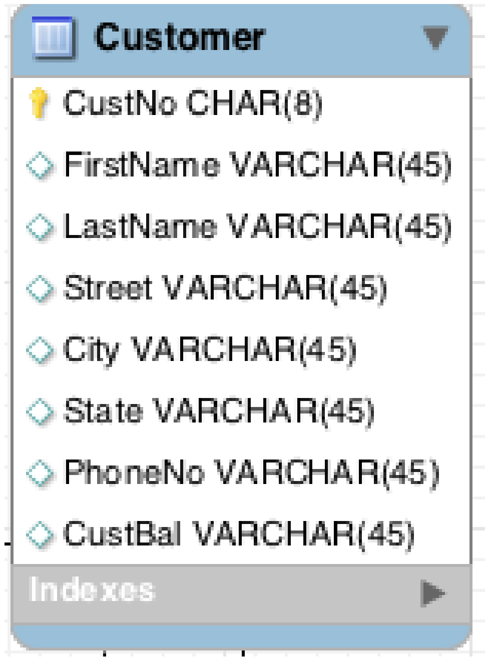
```
DROP [TEMPORARY] TABLE [IF EXISTS]
    tbl_name [, tbl_name] ...
    [RESTRICT | CASCADE]
```

Creating Tables

Example (no foreign key):



```
CREATE TABLE IF NOT EXISTS `Customer` (
  `CustNo`    CHAR(8) NOT NULL,
  `FirstName` VARCHAR(45) NULL,
  `LastName` VARCHAR(45) NULL,
  `Street` VARCHAR(45) NULL,
  `City` VARCHAR(45) NULL,
  `State` VARCHAR(45) NULL,
  `PhoneNo` VARCHAR(45) NULL,
  `CustBal` VARCHAR(45) NULL,
  PRIMARY KEY (`CustNo`))
ENGINE = InnoDB;
```



Introduction to InnoDB

- InnoDB is a general-purpose storage engine that balances high reliability and high performance. In MySQL 5.7, InnoDB is the default MySQL storage engine. Unless you have configured a different default storage engine, issuing a CREATE TABLE statement without an ENGINE= clause creates an InnoDB table.

```
-- Default storage engine = InnoDB.
CREATE TABLE t1 (a INT, b CHAR (20), PRIMARY KEY (a));
-- Backward-compatible with older MySQL.
CREATE TABLE t2 (a INT, b CHAR (20), PRIMARY KEY (a))
ENGINE=InnoDB;
```

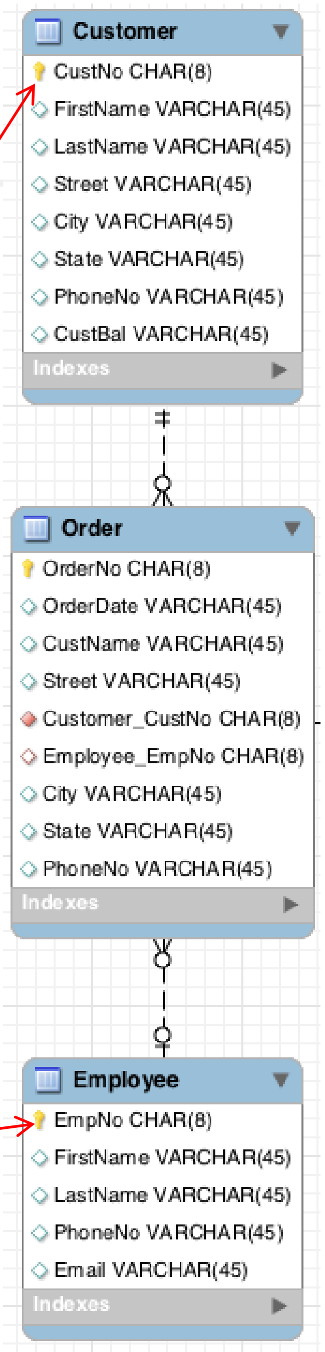
References:

- http://dev.mysql.com/doc/refman/5.7/en/innodb-introduction.html
- http://dev.mysql.com/doc/refman/5.7/en/using-innodb-tables.html

Creating Tables

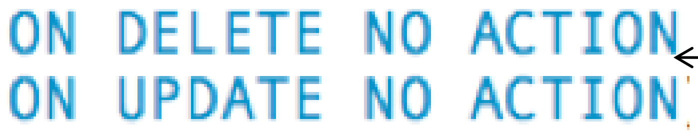
Example (with foreign keys):

```
CREATE TABLE IF NOT EXISTS `Order` (
  `OrderNo` CHAR(8) NOT NULL,
  `OrderDate` VARCHAR(45) NULL,
  `Customer_CustNo` CHAR(8) NOT NULL,
  `Employee_EmpNo` CHAR(8) NULL,
  `CustName` VARCHAR(45) NULL,
  `Street` VARCHAR(45) NULL,
  `City` VARCHAR(45) NULL,
  `State` VARCHAR(45) NULL,
  `PhoneNo` VARCHAR(45) NULL,
  PRIMARY KEY (`OrderNo`),
  INDEX `fk_Order_Customer_idx` (`Customer_CustNo` ASC),
  INDEX `fk_Order_Employee1_idx` (`Employee_EmpNo` ASC),
  CONSTRAINT `fk_Order_Customer`
    FOREIGN KEY (`Customer_CustNo`)
    REFERENCES `Customer` (`CustNo`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Order_Employee1`
    FOREIGN KEY (`Employee_EmpNo`)
    REFERENCES `Employee` (`EmpNo`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```



Constraints on Foreign Keys

In the previous example:



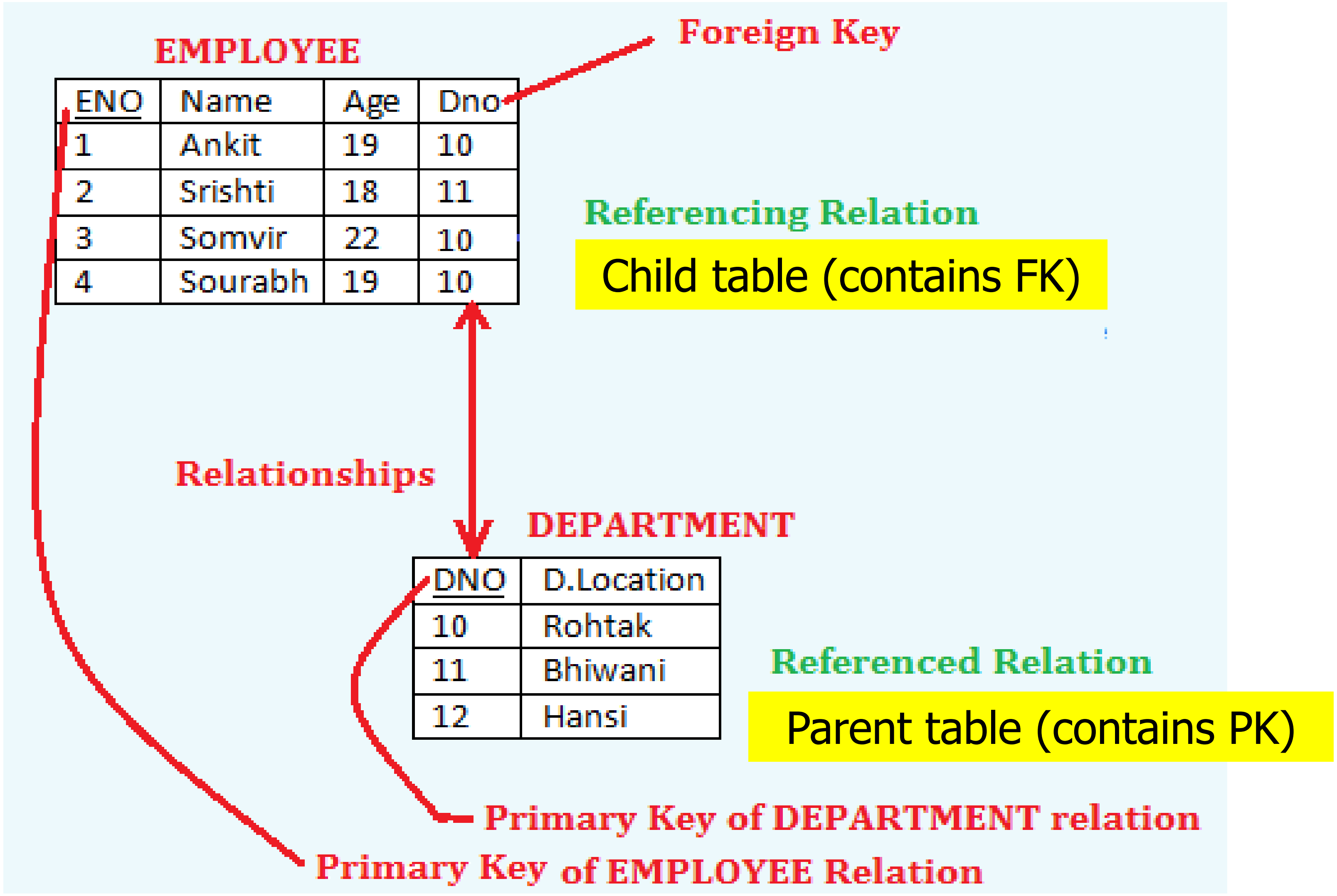
```
ON DELETE NO ACTION
ON UPDATE NO ACTION
```

Reference Option

The reference option can be one of the following:

1. **NO ACTION**: Server rejects the delete or update operation for the parent table if there is a related foreign key value in the **child** table.
2. **RESTRICT**: Rejects the delete or update operation for the parent table.
3. **CASCADE**: Delete or update the row from the parent table, and automatically delete or update the matching rows in the child table.
4. **SET NULL**: Delete or update the row from the parent table, and set the foreign key column or columns in the child table to NULL.

Constraints on Foreign Keys



EMPLOYEE
Foreign Key
Referencing Relation
Child table (contains FK)
Relationships
DEPARTMENT
Referenced Relation
Parent table (contains PK)
Primary Key of DEPARTMENT relation
Primary Key of EMPLOYEE Relation

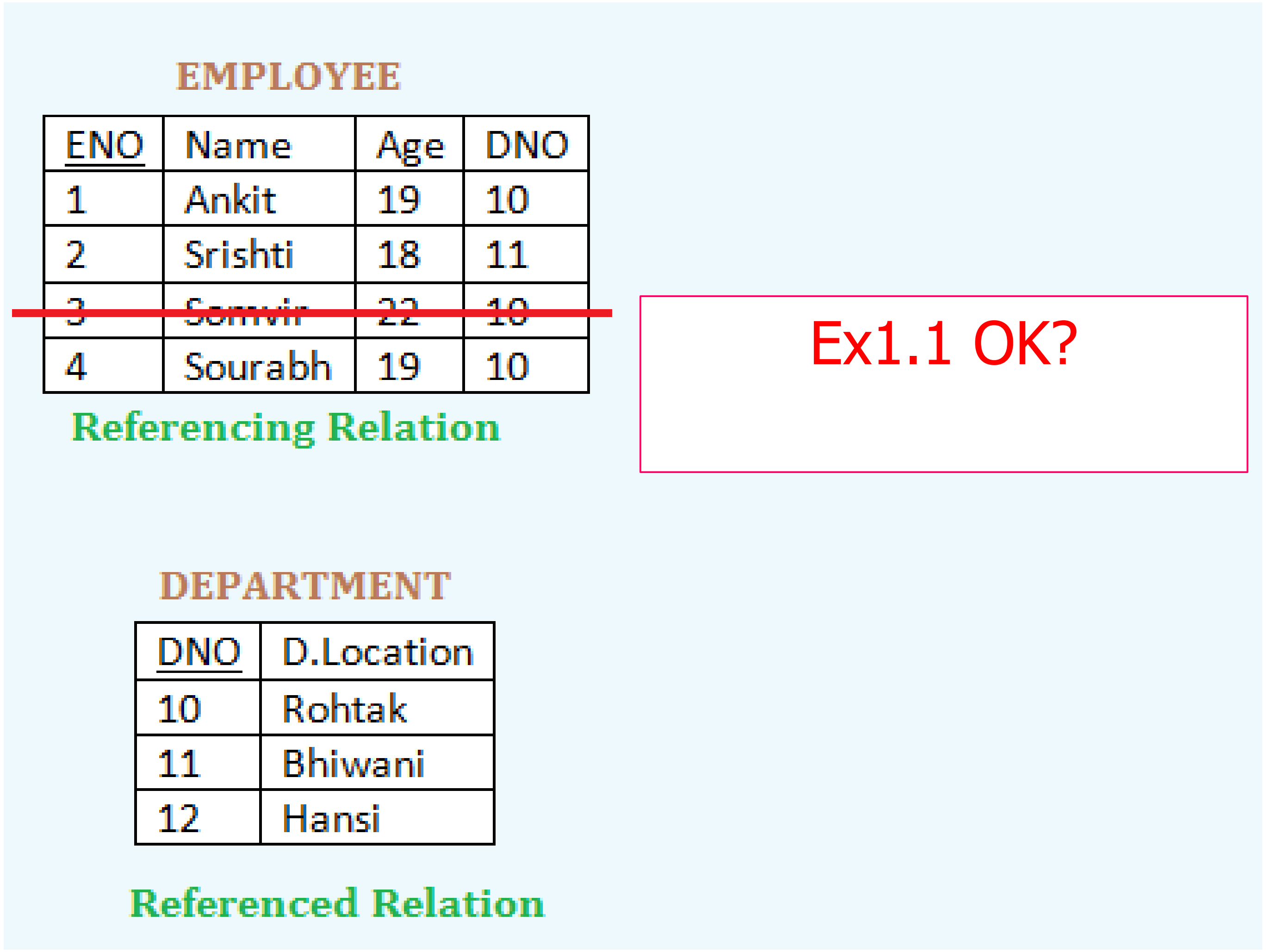
EMPLOYEE

ENO	Name	Age	Dno
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	10
4	Sourabh	19	10

DEPARTMENT

DNO	D.Location
10	Rohtak
11	Bhiwani
12	Hansi

1. Deletion in a Referencing Relation (EMPLOYEE relation)



EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
~~3~~	~~Somvir~~	~~22~~	~~10~~
4	Sourabh	19	10

Referencing Relation

Ex1.1 OK?

DEPARTMENT

DNO	D.Location
10	Rohtak
11	Bhiwani
12	Hansi

Referenced Relation

2. Deletion in a Referenced Relation (DEPARTMENT relation)

Ex 2.1

ON DELETE NO ACTION

DEPARTMENT

DNO	D.Location
~~10~~	~~Rohtak~~
11	Bhiwani
12	Hansi

EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	10
4	Sourabh	19	10

Ex 2.2

ON DELETE CASCADE

DEPARTMENT

DNO	D.Location
~~10~~	~~Rohtak~~
11	Bhiwani
12	Hansi

EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	10
4	Sourabh	19	10

Ex 2.3

ON DELETE SET NULL

DEPARTMENT

DNO	D.Location
~~10~~	~~Rohtak~~
11	Bhiwani
12	Hansi

EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	10
4	Sourabh	19	10

3. Update in a Referencing Relation (EMPLOYEE relation)

Ex 3.1

EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	~~10~~ 14
4	Sourabh	19	10

OK?

Referencing Relation

DEPARTMENT

DNO	D.Location
10	Rohtak
11	Bhiwani
12	Hansi

Referenced Relation

Ex 3.2

EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	~~10~~ 12
4	Sourabh	19	10

OK?

Referencing Relation

DEPARTMENT

DNO	D.Location
10	Rohtak
11	Bhiwani
12	Hansi

Referenced Relation

4. Update in a Referenced Relation (DEPARTMENT relation)

Ex 4.1

ON UPDATE NO ACTION

DEPARTMENT

DNO	D.Location
14 ~~10~~	Rohtak
11	Bhiwani
12	Hansi

EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	10
4	Sourabh	19	10

Ex 4.2

ON UPDATE CASCADE

DEPARTMENT

DNO	D.Location
14 ~~10~~	Rohtak
11	Bhiwani
12	Hansi

EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	10
4	Sourabh	19	10

Ex 4.3

ON UPDATE SET NULL

DEPARTMENT

DNO	D.Location
14 ~~10~~	Rohtak
11	Bhiwani
12	Hansi

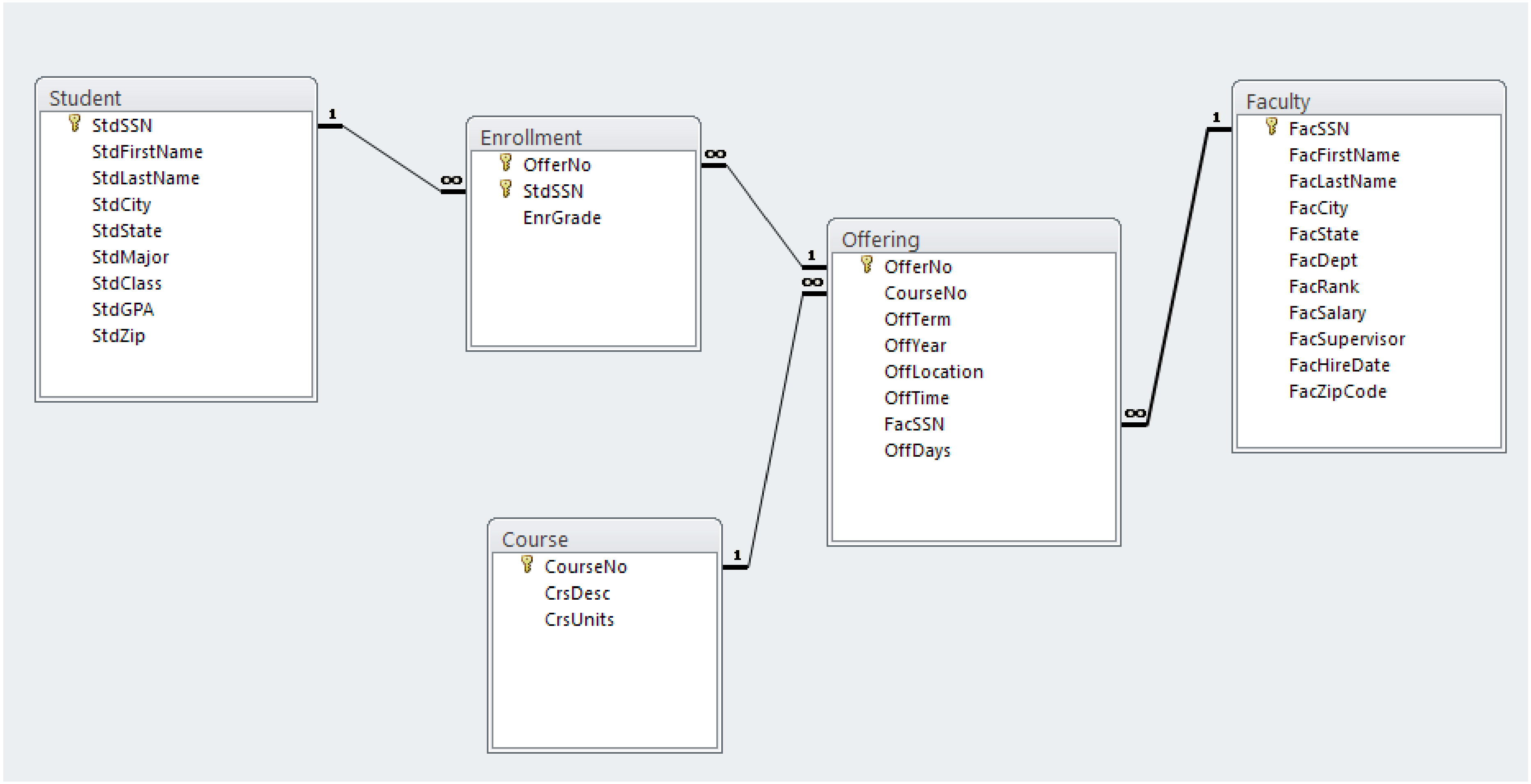
EMPLOYEE

ENO	Name	Age	DNO
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	10
4	Sourabh	19	10

SELECT Statement: Retrieval

SELECT <list of column expressions>
FROM <list of tables and join operations>
WHERE <list of logical expressions for <ins>rows</ins>>
GROUP BY <list of grouping columns>
HAVING <list of logical expressions for <ins>groups</ins>>
ORDER BY <list of sorting specifications>

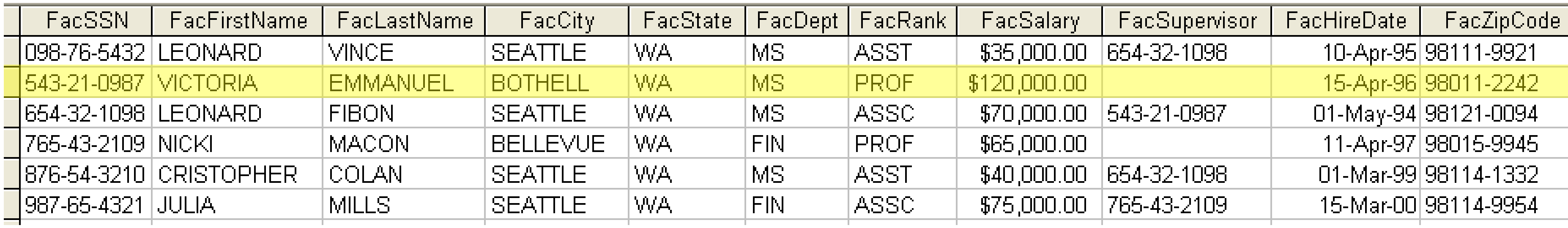
Example DB


Student
StdSSN
StdFirstName
StdLastName
StdCity
StdState
StdMajor
StdClass
StdGPA
StdZip
Enrollment
OfferNo
StdSSN
EnrGrade
Offering
OfferNo
CourseNo
OffTerm
OffYear
OffLocation
OffTime
FacSSN
OffDays
Faculty
FacSSN
FacFirstName
FacLastName
FacCity
FacState
FacDept
FacRank
FacSalary
FacSupervisor
FacHireDate
FacZipCode
Course
CourseNo
CrsDesc
CrsUnits
1
∞
1
∞
1
∞
1
∞

Single Table Problems

- Retrieves all rows and columns

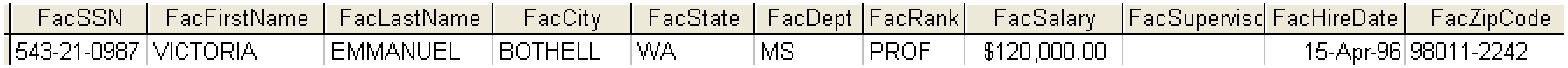
```
SELECT * FROM Faculty
```



FacSSN	FacFirstName	FacLastName	FacCity	FacState	FacDept	FacRank	FacSalary	FacSupervisor	FacHireDate	FacZipCode
098-76-5432	LEONARD	VINCE	SEATTLE	WA	MS	ASST	$35,000.00	654-32-1098	10-Apr-95	98111-9921
543-21-0987	VICTORIA	EMMANUEL	BOTHELL	WA	MS	PROF	$120,000.00		15-Apr-96	98011-2242
654-32-1098	LEONARD	FIBON	SEATTLE	WA	MS	ASSC	$70,000.00	543-21-0987	01-May-94	98121-0094
765-43-2109	NICKI	MACON	BELLEVUE	WA	FIN	PROF	$65,000.00		11-Apr-97	98015-9945
876-54-3210	CRISTOPHER	COLAN	SEATTLE	WA	MS	ASST	$40,000.00	654-32-1098	01-Mar-99	98114-1332
987-65-4321	JULIA	MILLS	SEATTLE	WA	FIN	ASSC	$75,000.00	765-43-2109	15-Mar-00	98114-9954

- Retrieves a subset of rows (restrict)

```
SELECT * FROM Faculty
WHERE FacSSN=‘543-21-0987’
```

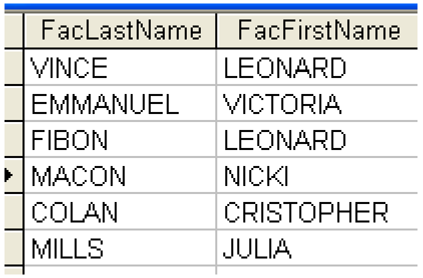


FacSSN	FacFirstName	FacLastName	FacCity	FacState	FacDept	FacRank	FacSalary	FacSupervisc	FacHireDate	FacZipCode
543-21-0987	VICTORIA	EMMANUEL	BOTHELL	WA	MS	PROF	$120,000.00		15-Apr-96	98011-2242

Single Table Problems

- Retrieves a subset of columns (project)

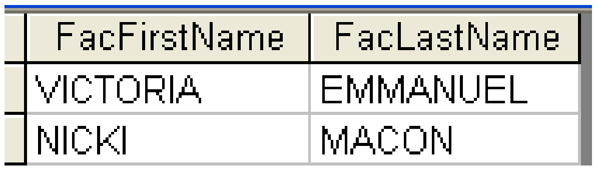
```
SELECT   FacLastName,
         FacFirstName
FROM     Faculty
```



FacLastName	FacFirstName
VINCE	LEONARD
EMMANUEL	VICTORIA
FIBON	LEONARD
MACON	NICKI
COLAN	CRISTOPHER
MILLS	JULIA

- Retrieves a subset of rows and columns

```
SELECT   FacFirstName, FacLastName
FROM     Faculty
WHERE    FacSalary >= 65000   AND
         FacRank='PROF'
```

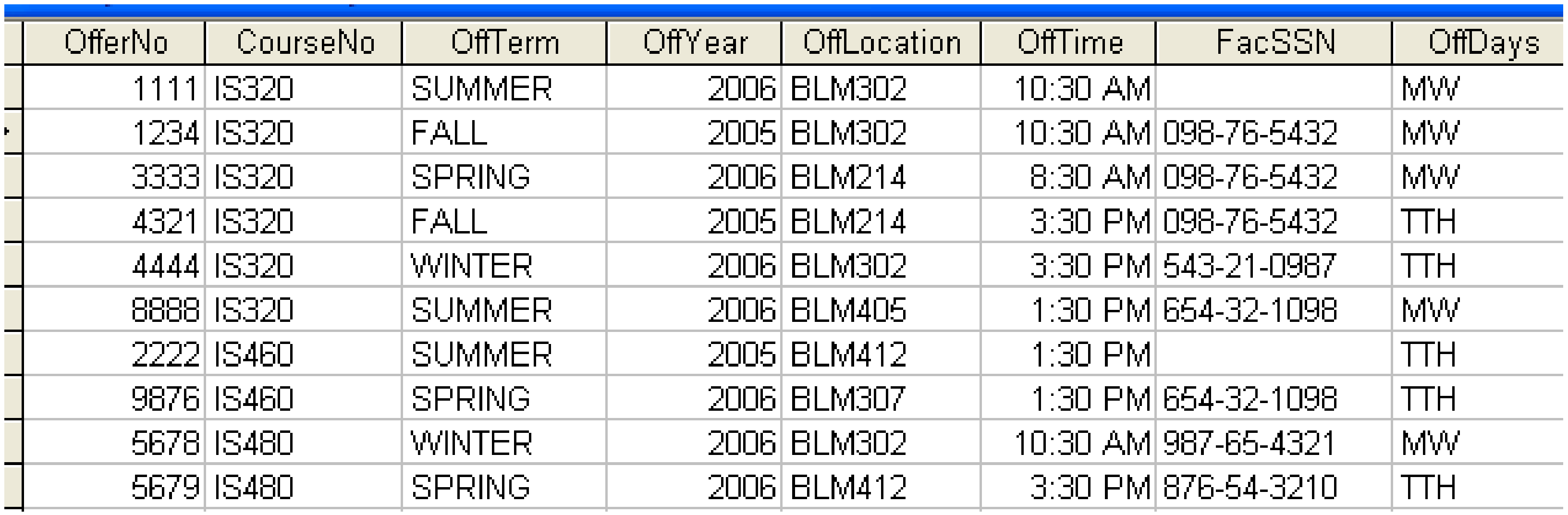


FacFirstName	FacLastName
VICTORIA	EMMANUEL
NICKI	MACON

Single Table Problems

- Inexact matching:
  - Match against a pattern: `LIKE` operator
  - Specify patterns: wildcard (%), single character (_)

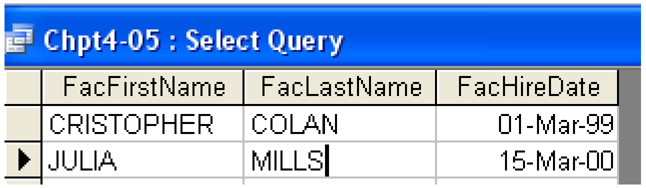
```
SELECT * FROM Offering
WHERE    CourseNo LIKE 'IS%'
```



OfferNo	CourseNo	OffTerm	OffYear	OffLocation	OffTime	FacSSN	OffDays
1111	IS320	SUMMER	2006	BLM302	10:30 AM		MW
1234	IS320	FALL	2005	BLM302	10:30 AM	098-76-5432	MW
3333	IS320	SPRING	2006	BLM214	8:30 AM	098-76-5432	MW
4321	IS320	FALL	2005	BLM214	3:30 PM	098-76-5432	TTH
4444	IS320	WINTER	2006	BLM302	3:30 PM	543-21-0987	TTH
8888	IS320	SUMMER	2006	BLM405	1:30 PM	654-32-1098	MW
2222	IS460	SUMMER	2005	BLM412	1:30 PM		TTH
9876	IS460	SPRING	2006	BLM307	1:30 PM	654-32-1098	TTH
5678	IS480	WINTER	2006	BLM302	10:30 AM	987-65-4321	MW
5679	IS480	SPRING	2006	BLM412	3:30 PM	876-54-3210	TTH

Single Table Problems

- Dates: are numbers

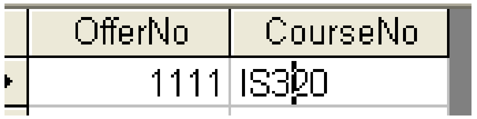


Chpt4-05 : Select Query

FacFirstName	FacLastName	FacHireDate
CRISTOPHER	COLAN	01-Mar-99
JULIA	MILLS	15-Mar-00

```
SELECT    FacFirstName, FacLastName
FROM      Faculty
WHERE     FacHireDate   BETWEEN
‘1999-1-1’ AND ’2000-12-31’
```

- Testing for null values



OfferNo	CourseNo
1111	IS320

```
SELECT    OfferNo, CourseN
FROM      Offering
WHERE     FacSSN IS NULL AND
          OffTerm=‘SUMMER’ AND
          OFFYear=2006;
```

Use of DISTINCT

List the property numbers of all properties that have been viewed.

```
SELECT propertyNo
FROM Viewing;
```

propertyNo
PA14
PG4
PG4
PA14
PG36

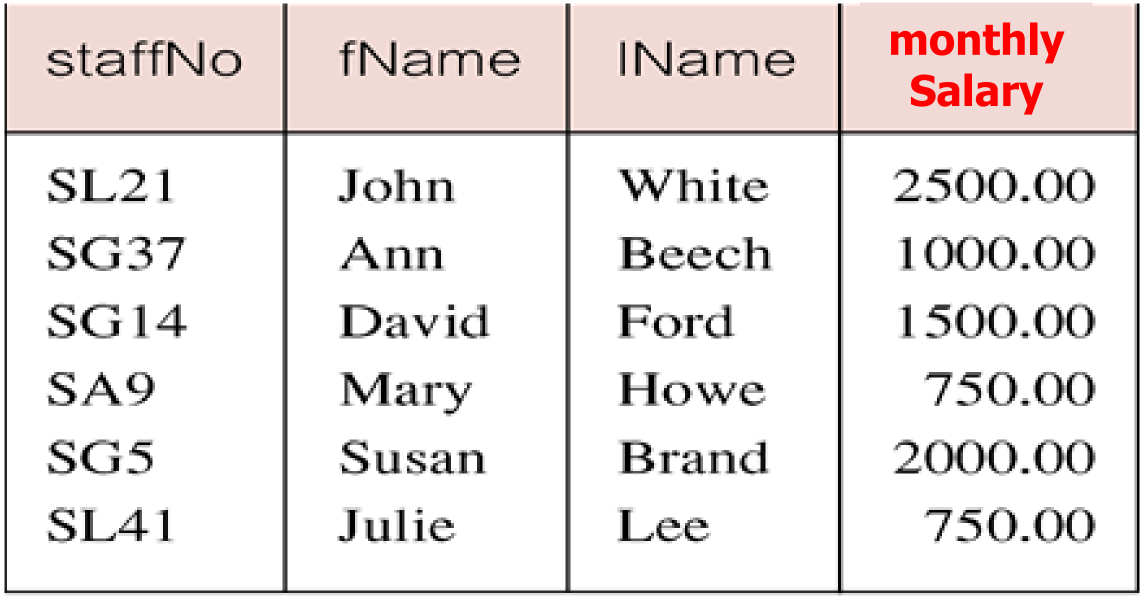
```
SELECT DISTINCT propertyNo
FROM Viewing;
```

propertyNo
PA14
PG4
PG36

Calculated Fields and Alias

Produce list of monthly salaries for all staff, showing staff number, first/last name, and salary.

```
SELECT staffNo, fName, lName, salary/12   AS monthlySalary
    FROM Staff;
```



staffNo	fName	lName	monthly Salary
SL21	John	White	2500.00
SG37	Ann	Beech	1000.00
SG14	David	Ford	1500.00
SA9	Mary	Howe	750.00
SG5	Susan	Brand	2000.00
SL41	Julie	Lee	750.00

Set Membership

List all managers and supervisors.

```
SELECT staffNo, fName, lName, position
FROM Staff
WHERE position IN (‘Manager’, ‘Supervisor’);
```

staffNo	fName	lName	position
SL21	John	White	Manager
SG14	David	Ford	Supervisor
SG5	Susan	Brand	Manager

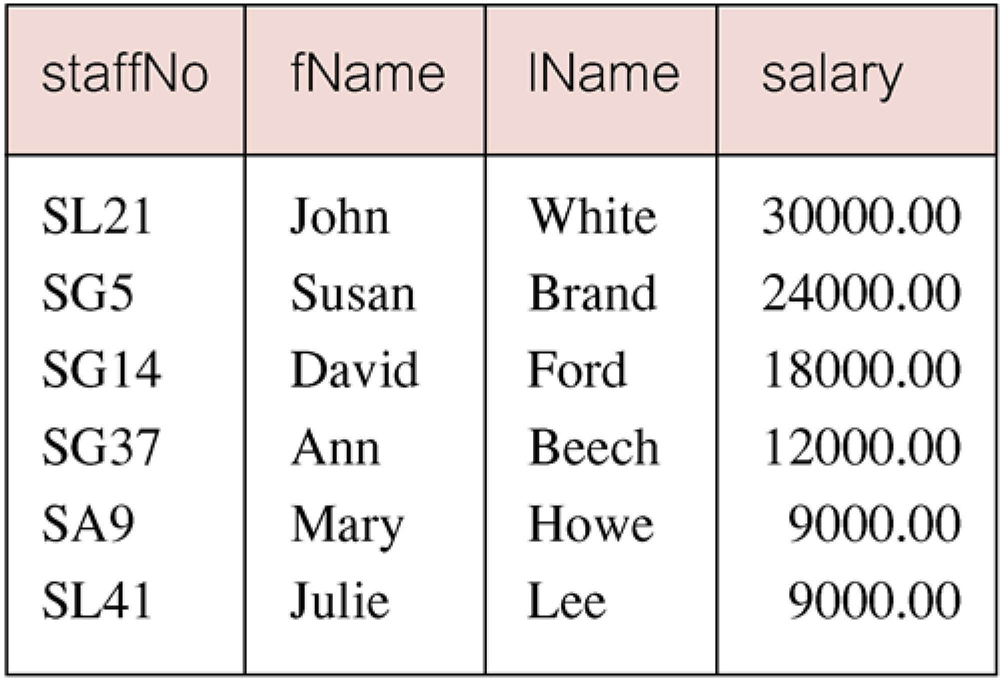
Ex 5

Same as:

Single Column Ordering

List salaries for all staff, arranged in descending order of salary.

```
SELECT staffNo, fName, lName, salary
    FROM Staff
    ORDER BY salary DESC;
```



staffNo	fName	lName	salary
SL21	John	White	30000.00
SG5	Susan	Brand	24000.00
SG14	David	Ford	18000.00
SG37	Ann	Beech	12000.00
SA9	Mary	Howe	9000.00
SL41	Julie	Lee	9000.00

Multiple Column Ordering

```
SELECT propertyNo, type, rooms, rent
        FROM PropertyForRent
        ORDER BY type;
```

propertyNo	type	rooms	rent
PL94	Flat	4	400
PG4	Flat	3	350
PG36	Flat	3	375
PG16	Flat	4	450
PA14	House	6	650
PG21	House	5	600

```
SELECT propertyNo, type, rooms, rent
        FROM PropertyForRent
        ORDER BY type, rent DESC;
```

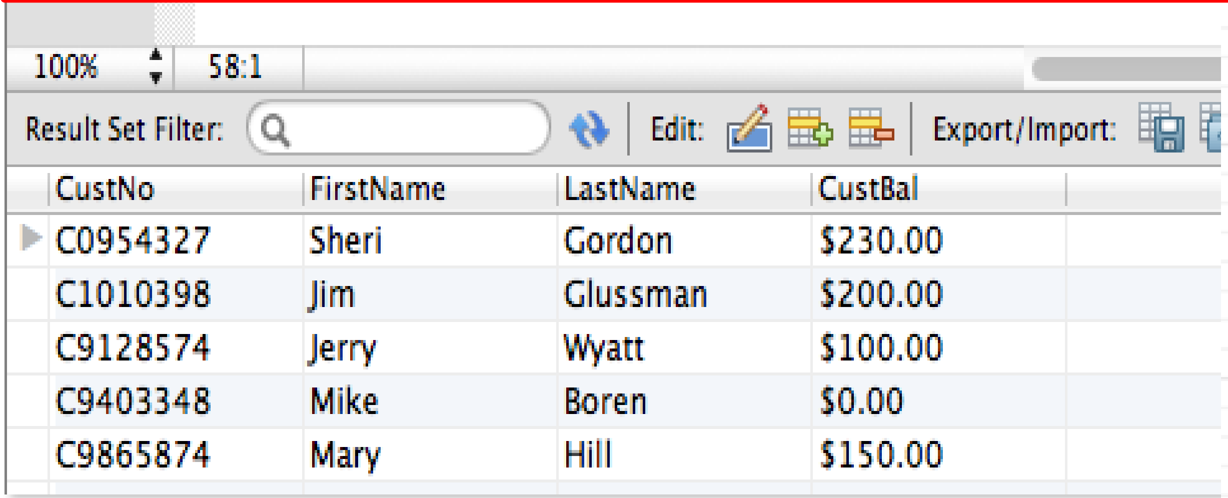
propertyNo	type	rooms	rent
PG16	Flat	4	450
PL94	Flat	4	400
PG36	Flat	3	375
PG4	Flat	3	350
PA14	House	6	650
PG21	House	5	600

SELECT from One Table

Ex 6.1 **Query:**

List the ID, name (first and last), and balance of customers who reside in Colorado (State is CO).

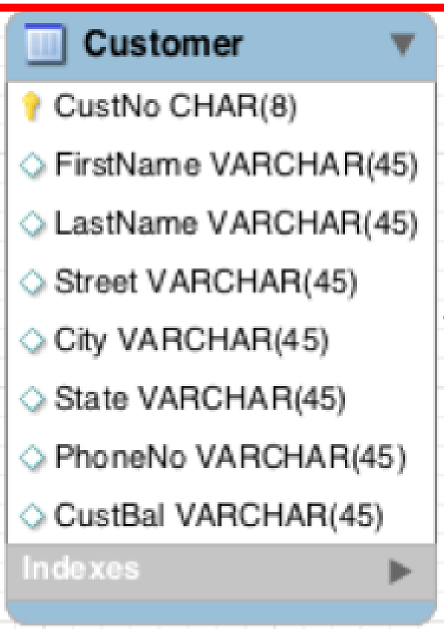
Solution:



100% 58:1

Result Set Filter: Edit: Export/Import:

CustNo	FirstName	LastName	CustBal
C0954327	Sheri	Gordon	$230.00
C1010398	Jim	Glussman	$200.00
C9128574	Jerry	Wyatt	$100.00
C9403348	Mike	Boren	$0.00
C9865874	Mary	Hill	$150.00



Customer
- CustNo CHAR(8)
- FirstName VARCHAR(45)
- LastName VARCHAR(45)
- Street VARCHAR(45)
- City VARCHAR(45)
- State VARCHAR(45)
- PhoneNo VARCHAR(45)
- CustBal VARCHAR(45)

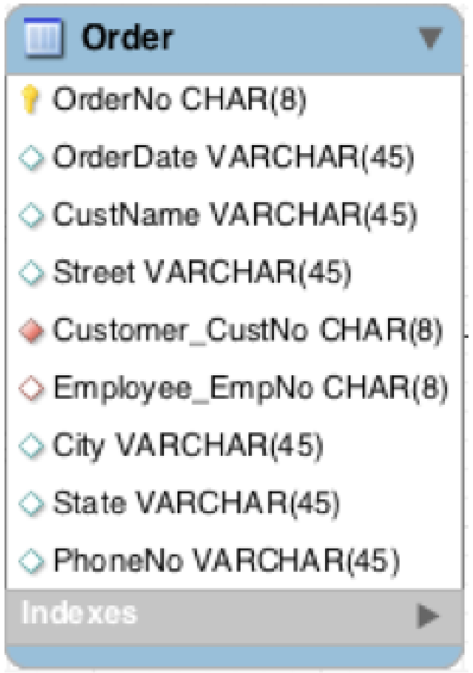
Indexes

SELECT from One Table

Ex 6.2 **Query:**

List the order number, order date, and customer (recipient) name of the orders that are sent to Denver, Englewood, or Hong Kong. Sort the result according to customer (recipient) name.

Solution:



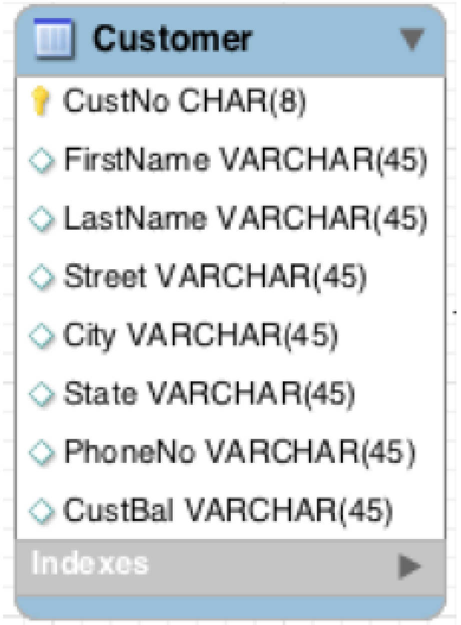
Order
OrderNo CHAR(8)
OrderDate VARCHAR(45)
CustName VARCHAR(45)
Street VARCHAR(45)
Customer_CustNo CHAR(8)
Employee_EmpNo CHAR(8)
City VARCHAR(45)
State VARCHAR(45)
PhoneNo VARCHAR(45)
Indexes

SELECT from One Table

Ex 6.3 **Query:**

List the customer number, the name (first and last), the city, and the balance of customers who reside in Seattle with a balance greater than $150 or who reside in Hong Kong with a balance greater than $50.

Solution:


Customer
CustNo CHAR(8)
FirstName VARCHAR(45)
LastName VARCHAR(45)
Street VARCHAR(45)
City VARCHAR(45)
State VARCHAR(45)
PhoneNo VARCHAR(45)
CustBal VARCHAR(45)

Data Manipulation Statements

- INSERT
  - Adds one or more rows
- UPDATE
  - Modifies one or more rows
  - Can use a WHERE clause
- DELETE
  - Removes one or more rows
  - Can use a WHERE clause

INSERT Example

- Insert a row into the Student table

```
INSERT INTO Student
(StdSSN, StdFirstName, StdLastName, StdClass,
   StdMajor, StdGPA)
VALUES
( ‘999999999’ , ‘JOE’ , “STEVE”, ‘FR’ , ‘IS’ ,0.0)
```

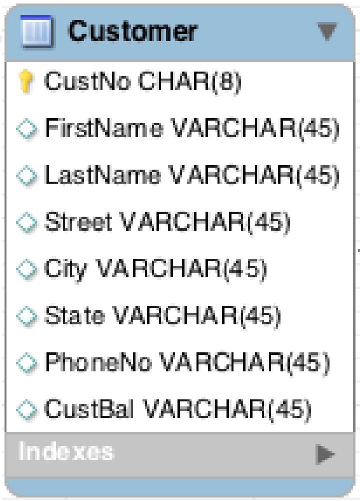
Example of INSERT

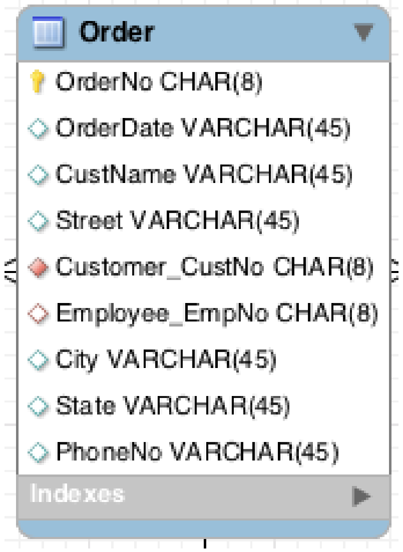
```
INSERT INTO `Customer`
VALUES ("C1234567","Man Wai","Mak","Yuk Choi Rd",
        "Hong Kong", "HK", "12345-6789","$100");

INSERT INTO `Order`
VALUES ("O1234567","5/25/2013","C1234567","E1329594",
        "Man-Wai Mak","Yuk Choi Rd.","Hong Kong","HK",
        "12345-6789");

INSERT INTO `Order`
VALUES ("O1111111","10/1/2007","C1234567","E8544399",
        "Man-Wai Mak","Yuk Choi Rd.","Hong Kong","HK",
        "12345-6789");

SELECT * FROM `Order` WHERE Customer_CustNo='C1234567';
```





OrderNo	OrderDate	Customer_CustNo	Employee_EmpNo	CustName	Street	City	State	PhoneNo
O1111111	10/1/2007	C1234567	E8544399	Man-Wai Mak	Yuk Choi Rd.	Hong Kong	HK	12345-6789
O1234567	5/25/2013	C1234567	E1329594	Man-Wai Mak	Yuk Choi Rd.	Hong Kong	HK	12345-6789

Update Example

- Change the major and class of “Homer Wells”

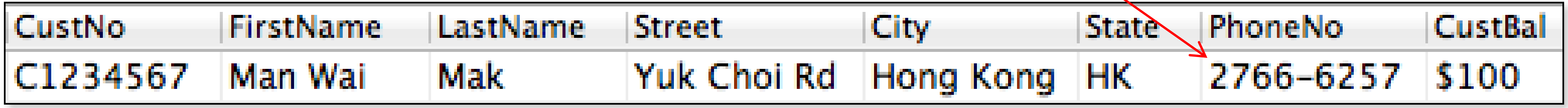
```
UPDATE Student
SET StdMajor = ‘ACCT’, StdClass= ‘SO’
WHERE StdFirstName= ‘HOMER’ AND
StdLastName= ‘WELLS’
```

Example of UPDATE

- Update the Phone number of a customer in `Customer` table.

```
UPDATE `Customer`
SET PhoneNo = '2766-6257'
WHERE LastName = 'Mak' AND FirstName = 'Man Wai';

SELECT * FROM `Customer` WHERE LastName = 'Mak';
```



CustNo	FirstName	LastName	Street	City	State	PhoneNo	CustBal
C1234567	Man Wai	Mak	Yuk Choi Rd	Hong Kong	HK	2766-6257	$100

Delete Example

- Delete all IS majors who are seniors

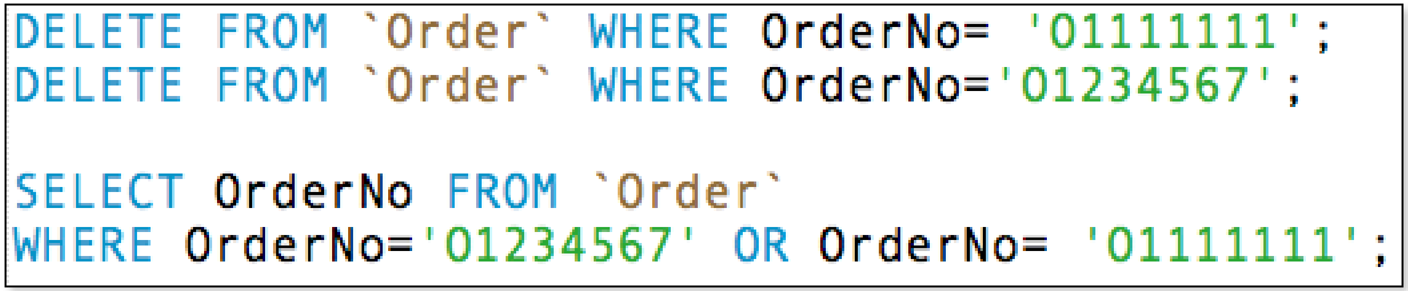
```
DELETE FROM Student
WHERE StdMajor = ‘IS’ AND StdClass= ‘SR’
```

- Delete all rows in a table

```
DELETE FROM Student
```

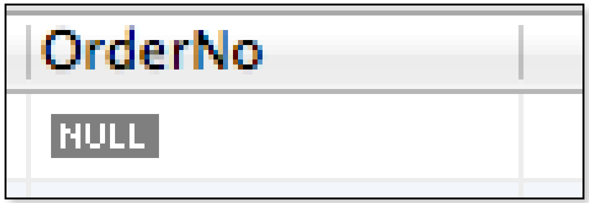
Example of DELETE

- Delete two orders in the `Order` table



```
DELETE FROM `Order` WHERE OrderNo= '01111111';
DELETE FROM `Order` WHERE OrderNo='01234567';

SELECT OrderNo FROM `Order`
WHERE OrderNo='01234567' OR OrderNo= '01111111';
```



OrderNo
NULL

References

- http://www.w3schools.com/sql/default.asp
- http://www.w3schools.com/sql/trysql.asp?filename=trysql_select_all
- http://www.w3schools.com/sql/sql_quiz.asp
- http://www.w3schools.com/sql/sql_quickref.asp
- http://www.sqlcourse.com
- http://www.sqlcourse2.com