Database Systems

Instructor: Bilal Khalid Dar

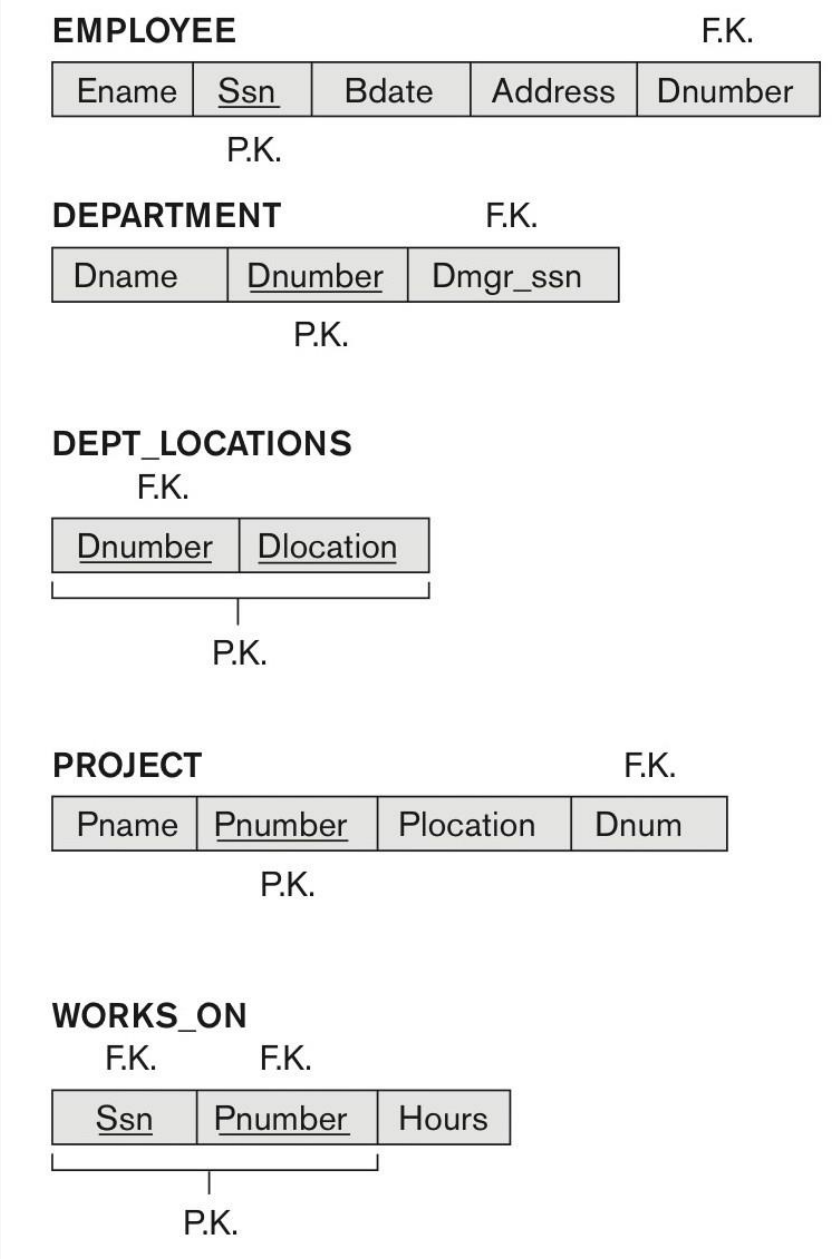
Agenda

- Normalization

Normalization Agenda

- We first discuss informal guidelines for good relational design
- Then we discuss formal concepts of functional dependencies and normal forms
  - - 1NF (First Normal Form)
  - - 2NF (Second Normal Form)
  - - 3NF (Third Normal Form)
  - - BCNF (Boyce-Codd Normal Form)

A simplified COMPANY relational database schema



Guidelines for good relational design

Semantics of the Relational Attributes must be clear

- GUIDELINE 1: Informally, each tuple in a relation should represent one entity or relationship instance. (Applies to individual relations and their attributes).
  - Attributes of different entities (EMPLOYEEs, DEPARTMENTs, PROJECTs) should not be mixed in the same relation
  - Only foreign keys should be used to refer to other entities
  - Entity and relationship attributes should be kept apart as much as possible.
- Bottom Line: *Design a schema that can be explained easily relation by relation. The semantics of attributes should be easy to interpret.*

Redundant Information in Tuples and Update Anomalies

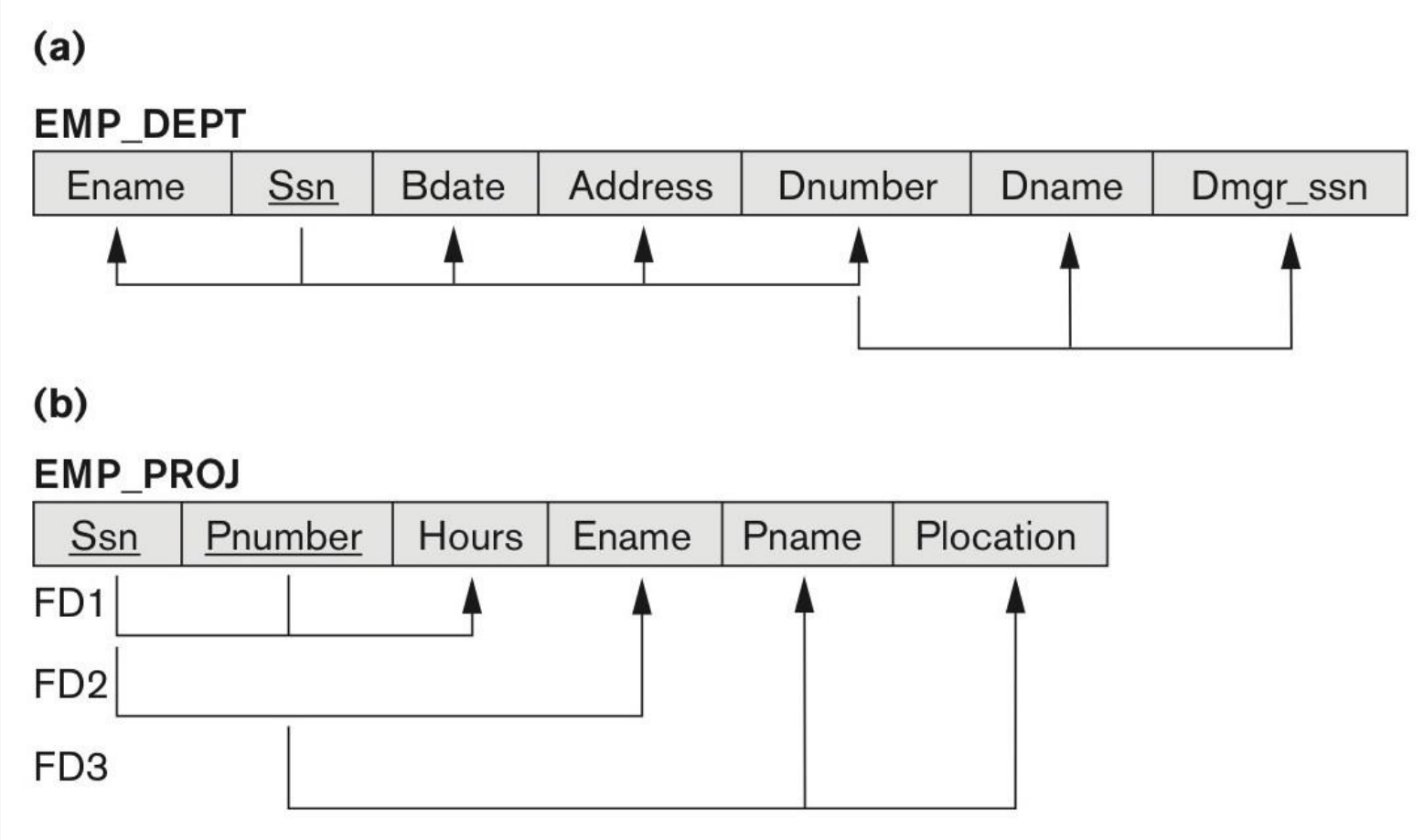
- Information is stored redundantly
  - Wastes storage
  - Causes problems with update anomalies
    - Insertion anomalies
    - Deletion anomalies
    - Modification anomalies

EXAMPLE OF AN UPDATE ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Update Anomaly:
  - Changing the name of project number P1 from “Billing” to “Customer-Accounting” may cause this update to be made for all 100 employees working on project P1.

Two relation schemas suffering from update anomalies

Figure 14.3
Two relation schemas suffering from update anomalies. (a) EMP_DEPT and (b) EMP_PROJ.



Sample states for EMP_DEPT and EMP_PROJ

Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

Sample states for EMP_DEPT and EMP_PROJ

Redundancy Redundancy

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

EXAMPLE OF AN INSERT ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Insert Anomaly:
  - Cannot insert a project unless an employee is assigned to it.
- Conversely
  - Cannot insert an employee unless an he/she is assigned to a project.

EXAMPLE OF A DELETE ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Delete Anomaly:
  - When a project is deleted, it will result in deleting all the employees who work on that project.
  - Alternately, if an employee is the sole employee on a project, deleting that employee would result in deleting the corresponding project.

Guideline for Redundant Information in Tuples and Update Anomalies

- GUIDELINE 2:
  - Design a schema that does not suffer from the insertion, deletion and update anomalies.
  - If there are any anomalies present, then note them so that applications can be made to take them into account.

Null Values in Tuples

- GUIDELINE 3:
  - Relations should be designed such that their tuples will have as few NULL values as possible
  - Attributes that are NULL frequently could be placed in separate relations (with the primary key)
- Reasons for nulls:
  - Attribute not applicable or invalid
  - Attribute value unknown (may exist)
  - Value known to exist, but unavailable

Generation of Spurious Tuples – avoid at any cost

- Bad designs for a relational database may result in erroneous results for certain JOIN operations
- The "lossless join" property is used to guarantee meaningful results for join operations

- GUIDELINE 4:
  - The relations should be designed to satisfy the lossless join condition.
  - No spurious tuples should be generated by doing a natural-join of any relations.

- **Before Normalization**
- Begin with a list of all of the fields that must appear in the database. Think of this as one big table.

– Do not include computed fields

– One place to begin getting this information is from a printed document used by the system.

– Additional attributes besides those for the entities described on the document can be added to the database.

Some Important Concepts

Functional Dependencies

Functional Dependencies

We say an attribute, B, has a *functional dependency* on another attribute, A, if for any two records, which have the same value for A, then the values for B in these two records must be the same. We illustrate this as:

A → B

Example: Suppose we keep track of employee email addresses, and we only track one email address for each employee. Suppose each employee is identified by their unique employee number. We say there is a functional dependency of email address on employee number:

employee number → email address

Functional Dependencies

EmpNum	EmpEmail	EmpFname	EmpLname
123	jdoe@abc.com	John	Doe
456	psmith@abc.com	Peter	Smith
555	alee1@abc.com	Alan	Lee
633	pdoe@abc.com	Peter	Doe
787	alee2@abc.com	Alan	Lee

If EmpNum is the PK then the FDs:

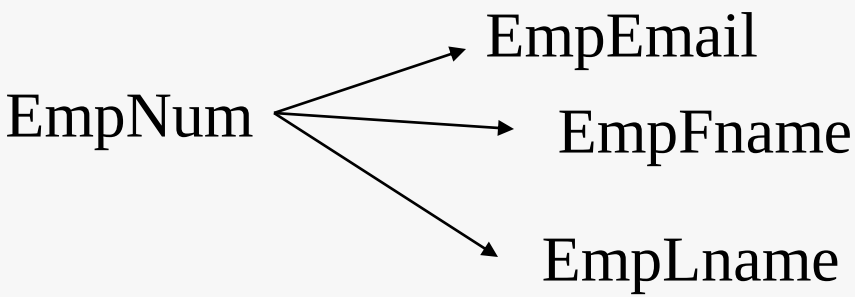
EmpNum → EmpEmail
EmpNum → EmpFname
EmpNum → EmpLname

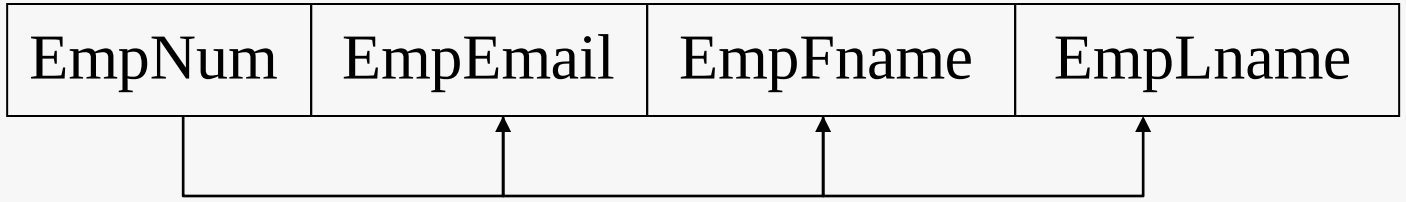
must exist.

Functional Dependencies

EmpNum → EmpEmail
EmpNum → EmpFname
EmpNum → EmpLname

3 different ways you might see FDs depicted





Determinant

Functional Dependency

EmpNum → EmpEmail

Attribute on the LHS is known as the ***determinant***

- EmpNum is a determinant of EmpEmail

Transitive dependency

Transitive dependency

Consider attributes A, B, and C, and where

A → B and B → C.

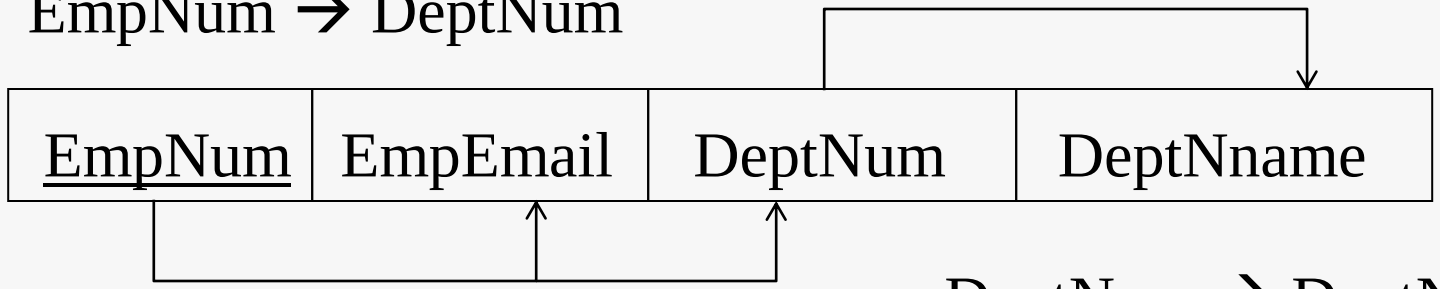
Functional dependencies are transitive, which means that we also have the functional dependency

A → C

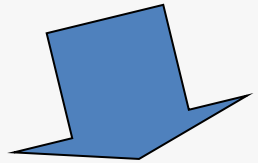
We say that C is transitively dependent on A through B.

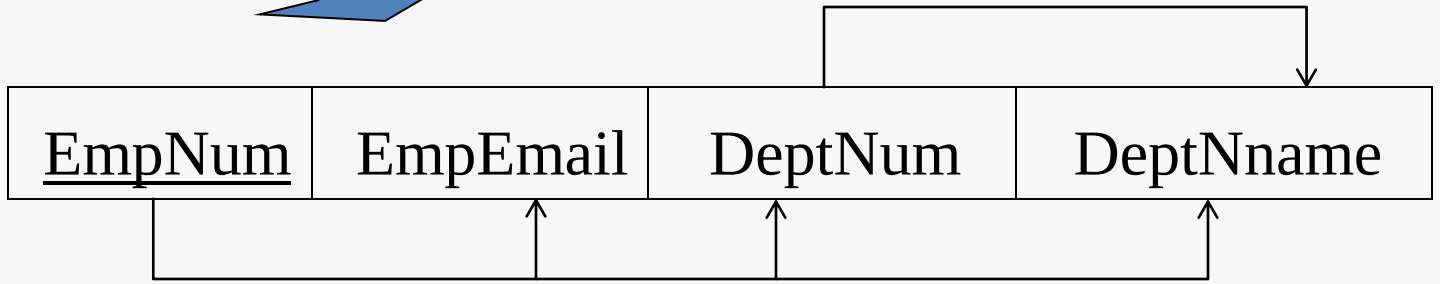
Transitive dependency

EmpNum → DeptNum



DeptNum → DeptName





DeptName is *transitively dependent* on EmpNum via DeptNum

EmpNum → DeptName

Normalization

Definition

- This is the process which allows you to winnow out redundant data within your database.
  - The results of a well executed normalization process are the same as those of a well planned E-R model
- This involves restructuring the tables to successively meeting higher forms of Normalization.
- A properly normalized database should have the following characteristics
  - Scalar values in each fields
  - Absence of redundancy.
  - Minimal use of null values.
  - Minimal loss of information.

(Note: Winnow(Webster): To get rid of / eliminate inferior material

Normalization

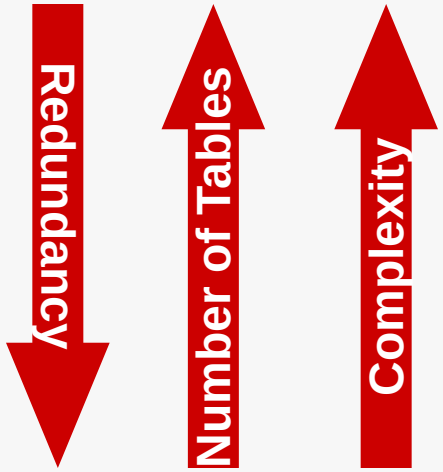
Process

- Eliminate Repeating Groups
  - Make a separate table for each set of related attributes and give each table a primary key.
- Eliminate Redundant Data
  - If an attribute depends on only part of a multivalued key, remove it to a separate table.
- Eliminate Columns not dependent on key
  - If attributes do not contribute to a description of the key, remove them to a separate table.
- Isolate Independent multiple relationships
  - No table may contain two or more 1:n or n:m relationships that are not directly related.
- Isolate Semantically Related Multiple Relationships
  - There may be practical constraints on information that justify separating logically related many-to-many relationships

Normalization

Levels

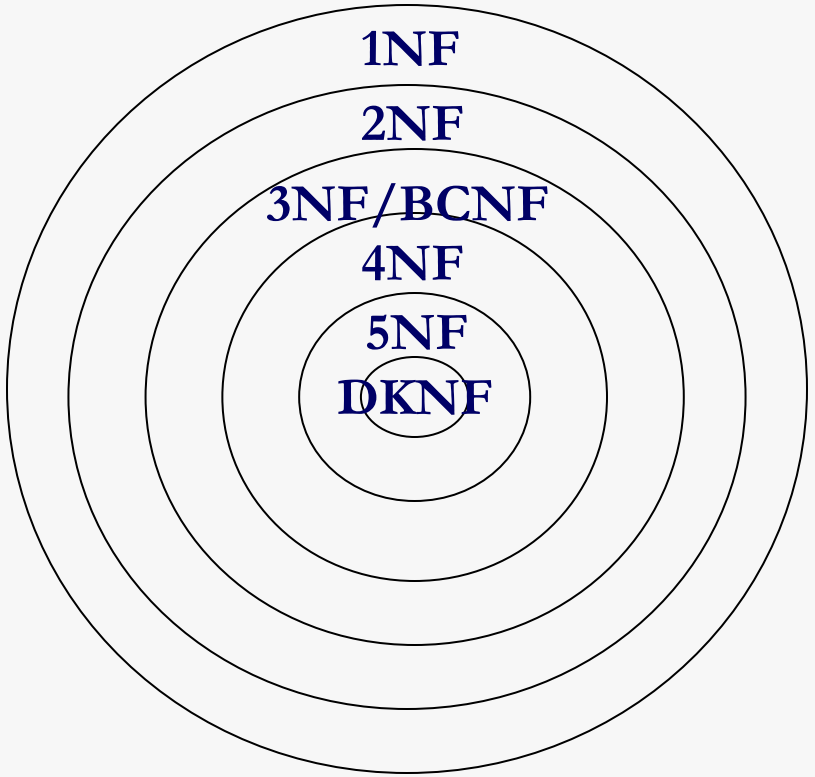
- Levels of normalization based on the amount of redundancy in the database.
- Relational theory defines a number of structure conditions called Normal Forms that assure that certain data anomalies do not occur in a database.
- Various levels of normalization are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - Third Normal Form (3NF)
  - Boyce-Codd Normal Form (BCNF)
  - Fourth Normal Form (4NF)
  - Fifth Normal Form (5NF)
  - Domain Key Normal Form (DKNF)



Most databases should be 3NF or BCNF in order to avoid the database anomalies.

Normalization

Levels



1NF	*Keys; No repeating groups or multi-valued*
2NF	*No partial dependencies*
3NF	*No transitive dependencies*
BCNF	*Determinants are candidate keys*
4NF	*No multivalued dependencies*
5NF	*No multivalued dependencies*
4NF	*No multivalued dependencies*

Each higher level is a subset of the lower level

Normalization

First Normal Form (1NF)

A table is considered to be in 1NF if all the fields contain only scalar values (as opposed to list of values).

Example (Not 1NF)

ISBN	Title	AuName	AuPhone	PubName	PubPhone	Price
0-321-32132-1	Balloon	Sleepy, Snoopy, Grumpy	321-321-1111, 232-234-1234, 665-235-6532	Small House	714-000-0000	$34.00
0-55-123456-9	Main Street	Jones, Smith	123-333-3333, 654-223-3455	Small House	714-000-0000	$22.95
0-123-45678-0	Ulysses	Joyce	666-666-6666	Alpha Press	999-999-9999	$34.00
1-22-233700-0	Visual Basic	Roman	444-444-4444	Big House	123-456-7890	$25.00

Author and AuPhone columns are not scalar

Normalization

1NF: Decomposition

1. Place all items appearing in the repeating group in a new table
2. Designate a primary key for each new table produced.
3. Create a relationship between the two tables
   - For 1:N relation duplicate the P.K. from 1 side to many side
   - For M:N relation create a new table with P.K. from both tables

Example (1NF)

ISBN	Title	PubName	PubPhone	Price
0-321-32132-1	Balloon	Small House	714-000-0000	$34.00
0-55-123456-9	Main Street	Small House	714-000-0000	$22.95
0-123-45678-0	Ulysses	Alpha Press	999-999-9999	$34.00
1-22-233700-0	Visual Basic	Big House	123-456-7890	$25.00

ISBN	AuName	AuPhone
0-321-32132-1	Sleepy	321-321-1111
0-321-32132-1	Snoopy	232-234-1234
0-321-32132-1	Grumpy	665-235-6532
0-55-123456-9	Jones	123-333-3333
0-55-123456-9	Smith	654-223-3455
0-123-45678-0	Joyce	666-666-6666
1-22-233700-0	Roman	444-444-4444

Normalization

Functional Dependencies

1. If one set of attributes in a table determines another set of attributes in the table, then the second set of attributes is said to be functionally dependent on the first set of attributes.

Example 1

ISBN	Title	Price
0-321-32132-1	Balloon	$34.00
0-55-123456-9	Main Street	$22.95
0-123-45678-0	Ulysses	$34.00
1-22-233700-0	Visual Basic	$25.00

Table Scheme: {ISBN, Title, Price}

Functional Dependencies: {ISBN} → {Title}

{ISBN} → {Price}

Normalization

Functional Dependencies

Example 2

PubID	PubName	PubPhone
1	Big House	999-999-9999
2	Small House	123-456-7890
3	Alpha Press	111-111-1111

Table Scheme: {PubID, PubName, PubPhone}
Functional Dependencies: {PubId} → {PubPhone}
{PubId} → {PubName}

Example 3

AuID	AuName	AuPhone
1	Sleepy	321-321-1111
2	Snoopy	232-234-1234
3	Grumpy	665-235-6532
4	Jones	123-333-3333
5	Smith	654-223-3455
6	Joyce	666-666-6666
7	Roman	444-444-4444

Table Scheme: {AuID, AuName, AuPhone}
Functional Dependencies: {AuId} → {AuPhone}
{AuId} → {AuName}

1NF Repeat

First Normal Form

First Normal Form

We say a relation is in **1NF** if all values stored in the relation are single-valued and atomic.

1NF places restrictions on the structure of relations. Values must be simple.

First Normal Form

The following in **not** in 1NF

EmpNum	**EmpPhone**	**EmpDegrees**
123	233-9876	
333	233-1231	BA, BSc, PhD
679	233-1231	BSc, MSc

EmpDegrees is a multi-valued field:

employee 679 has two degrees: *BSc* and *MSc*

employee 333 has three degrees: *BA*, *BSc*, *PhD*

First Normal Form

EmpNum	EmpPhone	EmpDegrees
123	233-9876	
333	233-1231	BA, BSc, PhD
679	233-1231	BSc, MSc

To obtain 1NF relations we must, without loss of information, replace the above with two relations - see next slide

First Normal Form

Employee

EmpNum	EmpPhone
123	233-9876
333	233-1231
679	233-1231

EmployeeDegree

EmpNum	EmpDegree
333	BA
333	BSc
333	PhD
679	BSc
679	MSc

Normalization

Second Normal Form (2NF)

For a table to be in 2NF, there are two requirements:

- The database is in first normal form
- All **nonkey** attributes in the table must be functionally dependent on the entire primary key

Note: *Remember that we are dealing with non-key attributes*

Example 1 (Not 2NF)

Scheme → {StudentId, CourseId, StudentName, CourseTitle, Grade}

1. **Key → {StudentId, CourseId}**
2. **{StudentId} → {StudentName}**
3. **{CourseId} → {CourseTitle}**
4. **{StudentId, CourseId} → {Grade}**
5. **StudentName depends on a subset of the key I.e. StudentId**
6. **CourseTitle depends on a subset of the key. i.e. CourseId**

Normalization

Second Normal Form (2NF)

Example 2 (Not 2NF)

Scheme → {City, Street, HouseNumber, HouseColor, CityPopulation}

1. **key → {City, Street, HouseNumber}**
2. **{City, Street, HouseNumber} → {HouseColor}**
3. **{City} → {CityPopulation}**
4. **CityPopulation does not belong to any key.**
5. **CityPopulation is functionally dependent on the City which is a proper subset of the key**

Example 3 (Not 2NF)

Scheme → {studio, movie, budget, studio_city}

1. **Key → {studio, movie}**
2. **{studio, movie} → {budget}**
3. **{studio} → {studio_city}**
4. **studio_city is not a part of a key**
5. **studio_city functionally depends on studio which is a proper subset of the key**

Normalization

2NF: Decomposition

1. If a data item is fully functionally dependent on only a part of the primary key, move that data item and that part of the primary key to a new table.
2. If other data items are functionally dependent on the same part of the key, place them in the new table also
3. Make the partial primary key copied from the original table the primary key for the new table.

   (Place all items that appear in the repeating group in a new table)

Example 1 (Convert to 2NF)

Old Scheme → {StudentId, CourseId, StudentName, CourseTitle, Grade}

New Scheme → {StudentId, StudentName}

New Scheme → {CourseId, CourseTitle}

New Scheme → {StudentId, CourseId, Grade}

Normalization

2NF: Decomposition

Example 2 (Convert to 2NF)

Old Scheme → {StudioID, Movie, Budget, StudioCity}

New Scheme → {Movie, StudioID, Budget}

New Scheme → {Studio, City}

Example 3 (Convert to 2NF)

Old Scheme → {City, Street, HouseNumber, HouseColor, CityPopulation}

New Scheme → {City, Street, HouseNumber, HouseColor}

New Scheme → {City, CityPopulation}

2NF Repeat

Second Normal Form

Second Normal Form

A relation is in **2NF** if it is in 1NF, and every non-key attribute is fully dependent on each candidate key. (That is, we don't have any partial functional dependency.)

- 2NF (and 3NF) both involve the concepts of key and non-key attributes.
- A *key attribute* is any attribute that is part of a key; any attribute that is not a key attribute, is a *non-key attribute*.
- Relations that are not in BCNF have data redundancies
- A relation in 2NF will not have any partial dependencies

Second Normal Form

Consider this **InvLine** table (in 1NF)**:**

InvNum	LineNum	ProdNum	Qty	InvDate

InvNum, LineNum ⟶ ProdNum, Qty

There are two candidate keys.

Qty is the only non-key attribute, and it is dependent on InvNum

InvNum ⟶ InvDate

Since there is a determinant that is not a candidate key, InvLine is **not BCNF**

InvLine is **not 2NF** since there is a partial dependency of InvDate on InvNum

InvLine is only in **1NF**

Second Normal Form

InvLine

InvNum	LineNum	ProdNum	Qty	InvDate

The above relation has redundancies: the invoice date is repeated on each invoice line.

We can *improve* the database by decomposing the relation into two relations:

InvNum	LineNum	ProdNum	Qty

InvNum	InvDate

Question: What is the highest normal form for these relations? 2NF? 3NF? BCNF?

Normalization

Third Normal Form (3NF)

- This form dictates that all **non-key** attributes of a table must be functionally dependent on a candidate key such that there are no interdependencies among non-key attributes i.e. there should be no transitive dependencies

- For a table to be in 3NF, there are two requirements
  - The table should be second normal form
  - No attribute is transitively dependent on the primary key

Example (Not in 3NF)

Scheme → {Title, PubID, BookType, Price }

1. **Key → {Title, PubId}**
2. **{Title, PubId} → {BookType}**
3. **{BookType} → {Price}**
4. **Both Price and BookType depend on a key hence 2NF**
5. **Transitively {Title, PubID} → {Price} hence not in 3NF**

Title	PubID	BookType	Price
Moby Dick	1	Adventure	34.95
Giant	2	Adventure	34.95
MobyDick	2	Adventure	34.95
Iliad	1	War	44.95
Romeo &Juliet	1	Love	59.90

Normalization

3NF: Decomposition

1. Move all items involved in transitive dependencies to a new entity.
2. Identify a primary key for the new entity.
3. Place the primary key for the new entity as a foreign key on the original entity.

Example 1 (Convert to 3NF)

Old Scheme → {Title, PubID, BookType, Price }

New Scheme → {BookType, Price}

New Scheme → {Title, PubID, BookType}

Normalization

3NF: Decomposition

Example 2 (Convert to 3NF)

Old Scheme → {BuildingID, Contractor, Fee}

New Scheme → {BuildingID, Contractor}

New Scheme → {Contractor, Fee}

BuildingID	Contractor
100	Randolph
150	Ingersoll
200	Randolph
250	Pitkin
300	Randolph

Contractor	Fee
Randolph	1200
Ingersoll	1100
Pitkin	1100

Normalization

Boyce-Codd Normal Form (BCNF)

- Consider following

StudID	Major	Advisor	MajGPA
123	Literature	Shakespeare	2.89
234	Music	Beethoven	4.00
345	Physics	Newton	3.76
456	Music	Mozart	2.23
567	Music	Beethoven	3.25
678	Physics	Bohr	2.56
789	Literature	Eliot	3.33

The table is in 3NF since no functional dependencies exist between two (or more) non-key attributes

- Major is NOT dependent on Advisor (Or vice versa)
- Major is NOT dependent on MajGPA (Or vice versa)
- Advisor is NOT dependent on MajGPA (Or vice versa)

Normalization

Boyce-Codd Normal Form (BCNF)

- Consider following

StudID	Major	Advisor	MajGPA
123	Literature	Shakespeare	2.89
234	Music	Beethoven	4.00
345	Physics	Newton	3.76
456	Music	Mozart	2.23
567	Music	Beethoven	3.25
678	Physics	Bohr	2.56
789	Literature	Eliot	3.33

HOWEVER, in the table, Beethoven appears twice

If Beethoven were to quit and be replaced by Dr. Dre, we would have to make TWO different changes

(A modification anomaly)

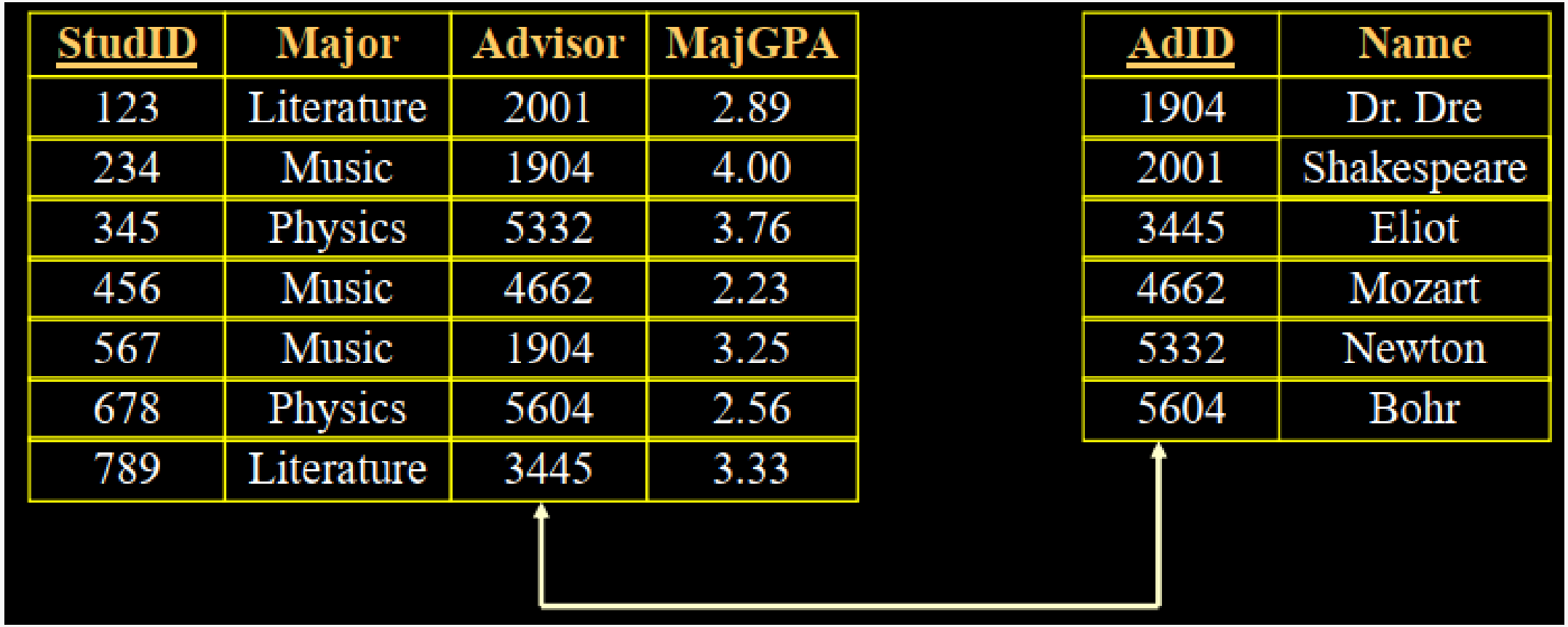
Normalization

Boyce-Codd Normal Form (BCNF)

- Solution, separate the two tables

StudID	Major	Advisor	MajGPA
123	Literature	2001	2.89
234	Music	1904	4.00
345	Physics	5332	3.76
456	Music	4662	2.23
567	Music	1904	3.25
678	Physics	5604	2.56
789	Literature	3445	3.33

AdID	Name
1904	Dr. Dre
2001	Shakespeare
3445	Eliot
4662	Mozart
5332	Newton
5604	Bohr



Third Normal Form

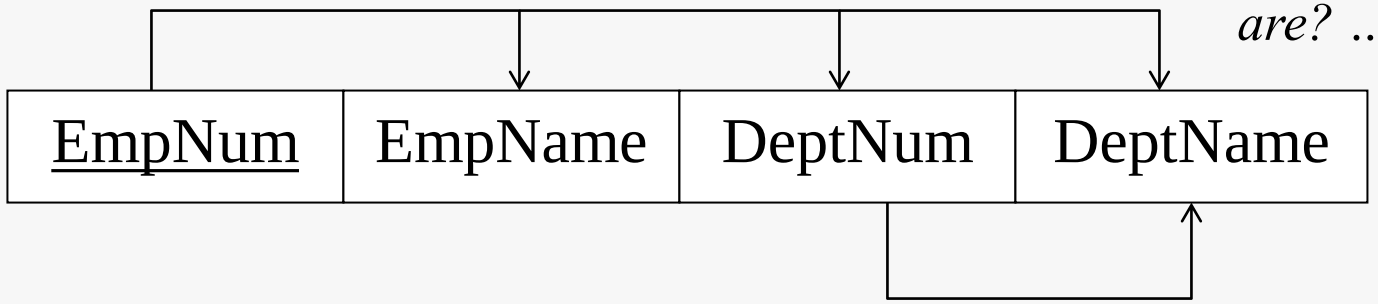
Third Normal Form

- A relation is in **3NF** if the relation is in 1NF and all determinants of *non-key* attributes are candidate keys
  That is, for any functional dependency: X $\rightarrow$ Y, where Y is a non-key attribute (or a set of non-key attributes), X is a candidate key.
- This definition of 3NF differs from BCNF only in the specification of non-key attributes - 3NF is weaker than BCNF. (BCNF requires all determinants to be candidate keys.)
- A relation in 3NF will not have any transitive dependencies of non-key attribute on a candidate key through another non-key attribute.

Third Normal Form

Consider this **Employee** relation

Candidate keys are? …



EmpName, DeptNum, and DeptName are non-key attributes.

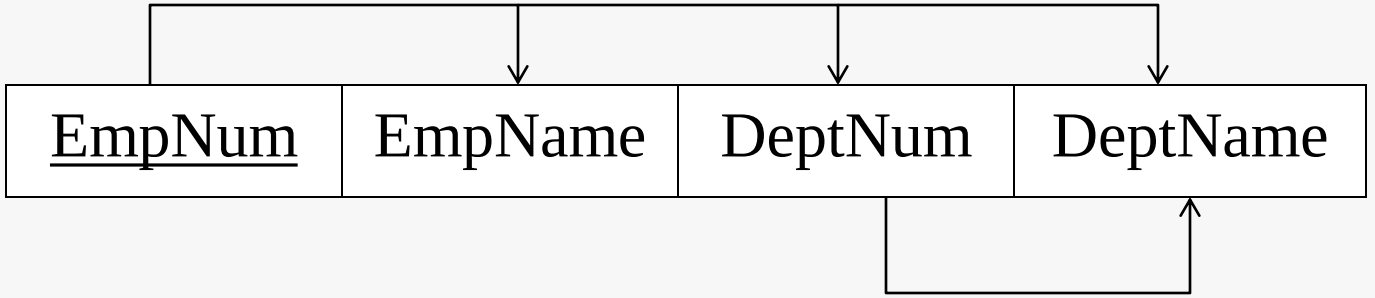
DeptNum determines DeptName, a non-key attribute, and DeptNum is not a candidate key.

Is the relation in 3NF? … no

Is the relation in BCNF? … no

Is the relation in 2NF? … yes

Third Normal Form



We correct the situation by decomposing the original relation into two 3NF relations. Note the decomposition is *lossless*.

Verify these two relations are in 3NF.

Normalization

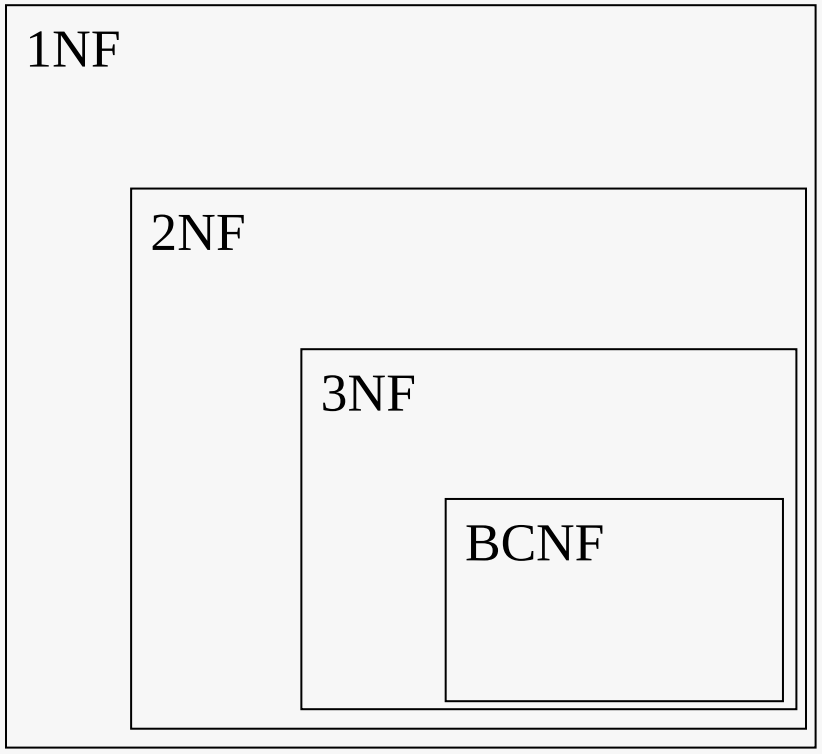
There is a sequence to normal forms:

1NF is considered the weakest,
2NF is stronger than 1NF,
3NF is stronger than 2NF, and
BCNF is considered the strongest

Also,

any relation that is in BCNF, is in 3NF;
any relation in 3NF is in 2NF; and
any relation in 2NF is in 1NF.

Normalization



a relation in BCNF, is also in 3NF

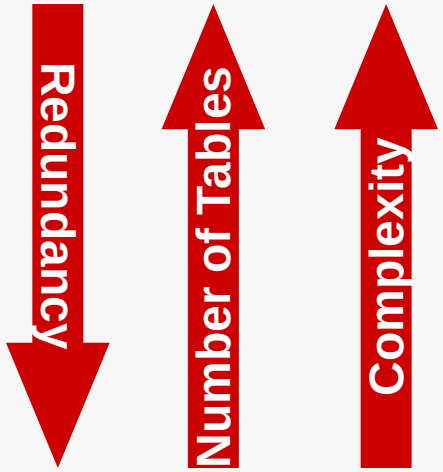
a relation in 3NF is also in 2NF

a relation in 2NF is also in 1NF

Normalization

Levels

- Levels of normalization based on the amount of redundancy in the database.
- Relational theory defines a number of structure conditions called Normal Forms that assure that certain data anomalies do not occur in a database.
- Various levels of normalization are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - Third Normal Form (3NF)
  - Boyce-Codd Normal Form (BCNF)
  - Fourth Normal Form (4NF)
  - Fifth Normal Form (5NF)
  - Domain Key Normal Form (DKNF)



Most databases should be 3NF or BCNF in order to avoid the database anomalies.

Normalization

Summary

- Different Stages of Normalization
  - **1NF** Keys; No repeating groups
  - **2NF** No partial dependencies
  - **3NF** No transitive dependencies
  - **BCNF** Determinants are candidate keys
  - **4NF** No multivalued dependencies
  - **5NF** Remove m-way relationships
  - **DKNF** Use domain constraints to enforce dependencies

Exercise – Convert the given schema in 3NF

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
15	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
15	Evergreen	101	John G. News	Database Designer	$105.00	19.4
15	Evergreen	105	Alice K. Johnson *	Database Designer	$105.00	35.7
15	Evergreen	106	William Smithfield	Programmer	$35.75	12.5
15	Evergreen	102	David H. Senior	Systems Analyst	$96.75	23.9
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
18	Amber Wave	118	James J. Frommer	General Support	$18.36	45.3
18	Amber Wave	104	Anne K. Ramoras *	Systems Analyst	$96.75	32.1
18	Amber Wave	112	Darlene M. Smithson	DSS Analyst	$45.95	44.0
22	Rolling Tide	105	Alice K. Johnson	Database Designer	$105.00	64.7
22	Rolling Tide	104	Anne K. Ramoras	Systems Analyst	$96.75	48.9
22	Rolling Tide	113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6
22	Rolling Tide	111	Geoff B. Wabash	Clerical Support	$26.87	22.5
22	Rolling Tide	106	William Smithfield	Programmer	$35.75	12.1
25	Starflight	107	Maria D. Alonzo	Programmer	$35.75	24.7
25	Starflight	115	Travis B. Bawangi	Systems Analyst	$96.75	45.8
25	Starflight	101	John G. News *	Database Designer	$105.00	56.3
25	Starflight	114	Annelise Jones	Applications Designer	$48.10	33.1


Thank
You