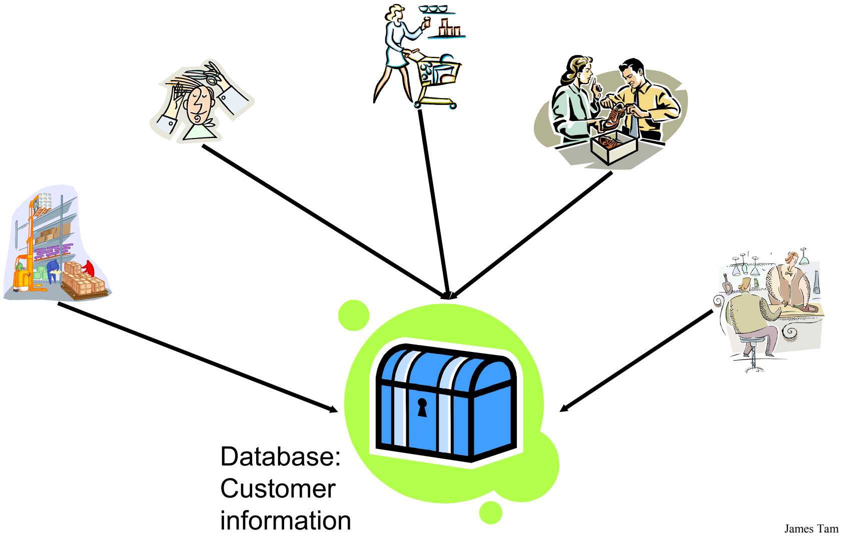
Databases

In this section of notes you will learn about: how information is stored in a database, the different types of relations that can exist within a database, how information can be retrieved via queries and principles for designing/redesigning databases.

Purpose Of A Database

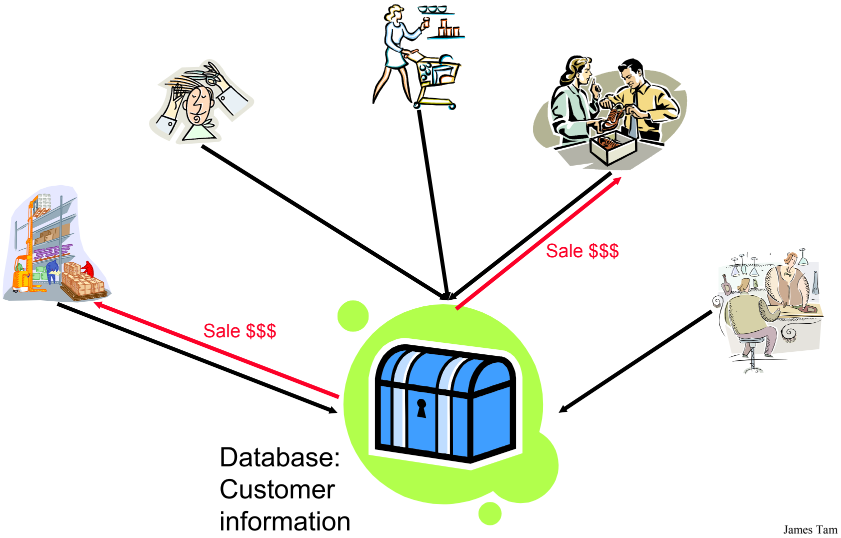
- To store information



Database:
Customer
information

Purpose Of A Database

- To retrieve information



Sale $$$

Sale $$$

Database: Customer information

With Bother With Databases?

- Are used to store and retrieve information
- Why bother, why not use a simple file as an alternative?
  - E.g., tracking client information

MILES EDWARD O'BRIAN
DS9 Corp
Electrical engineering
2007 purchases: $10,0000,000
2006 purchases: $1,750,000

JAMIE SMYTHE
Cooperative services
Gasoline refining
2006 purchases: $5,000,0000
2005 purchases: $5,000,0000
2004 purchases: $5,000,0000
2003 purchases: $5,000,0000
2002 purchases: $5,000,0000

SCOTT BRUCE
Bryce Consulting
Investment analysis
2007 purchases: $500,000
2006 purchases: $1,500,000
2005 purchases: $2,500,000
2004 purchases: $500,000

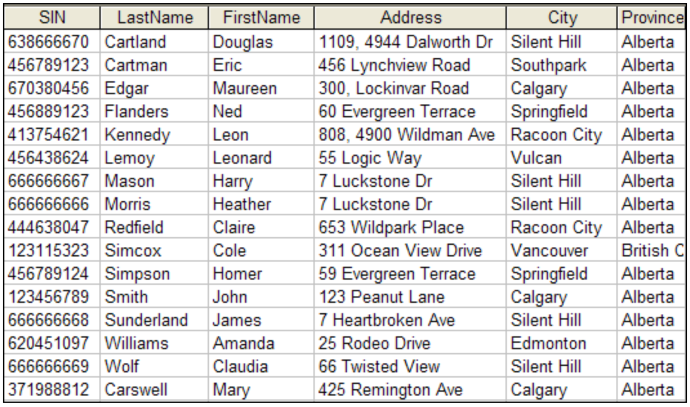
Etc.

- If the list is short then a simple text file may suffice.
- As the list grows organizing and updating the information becomes more challenging (duplicates or inaccuracies?)
- Validity must be manually checked.
- Also searching the list according to specific criteria may become difficult .
  - e.g., Show all clients whose purchases in 2007 were between one and five million dollars
  - e.g., Show all clients that made in one year a purchase exceeding 10 million dollars.

Storing Information In A Database

- Information is commonly stored in tables (relational database):

‘Employees’ table

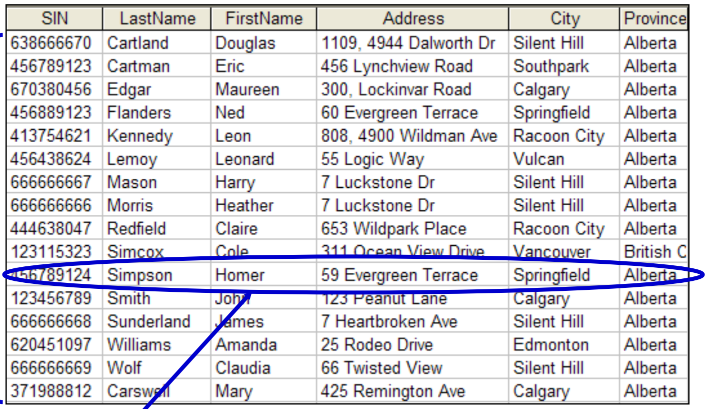


SIN	LastName	FirstName	Address	City	Province
638666670	Cartland	Douglas	1109, 4944 Dalworth Dr	Silent Hill	Alberta
456789123	Cartman	Eric	456 Lynchview Road	Southpark	Alberta
670380456	Edgar	Maureen	300, Lockinvar Road	Calgary	Alberta
456889123	Flanders	Ned	60 Evergreen Terrace	Springfield	Alberta
413754621	Kennedy	Leon	808, 4900 Wildman Ave	Racoon City	Alberta
456438624	Lemoy	Leonard	55 Logic Way	Vulcan	Alberta
666666667	Mason	Harry	7 Luckstone Dr	Silent Hill	Alberta
666666666	Morris	Heather	7 Luckstone Dr	Silent Hill	Alberta
444638047	Redfield	Claire	653 Wildpark Place	Racoon City	Alberta
123115323	Simcox	Cole	311 Ocean View Drive	Vancouver	British C
456789124	Simpson	Homer	59 Evergreen Terrace	Springfield	Alberta
123456789	Smith	John	123 Peanut Lane	Calgary	Alberta
666666668	Sunderland	James	7 Heartbroken Ave	Silent Hill	Alberta
620451097	Williams	Amanda	25 Rodeo Drive	Edmonton	Alberta
666666669	Wolf	Claudia	66 Twisted View	Silent Hill	Alberta
371988812	Carswell	Mary	425 Remington Ave	Calgary	Alberta

Storing Information In A Database (2)

- Record: An example instance of data within the table.

Records of the table

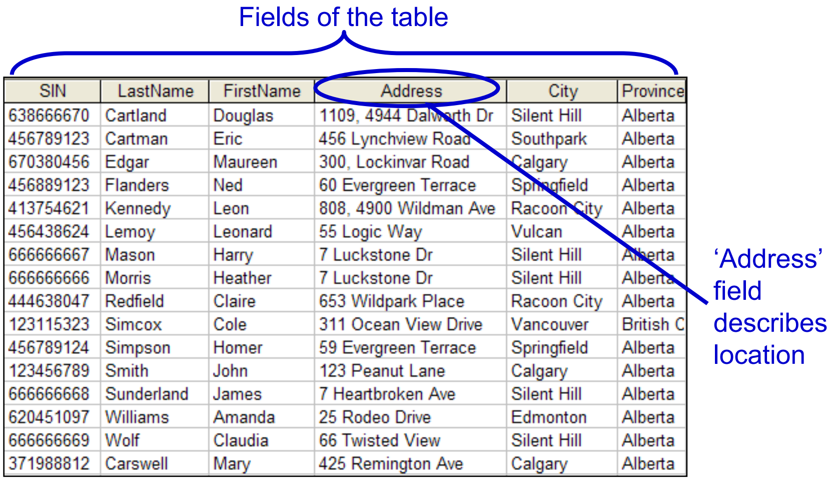


SIN	LastName	FirstName	Address	City	Province
638666670	Cartland	Douglas	1109, 4944 Dalworth Dr	Silent Hill	Alberta
456789123	Cartman	Eric	456 Lynchview Road	Southpark	Alberta
670380456	Edgar	Maureen	300, Lockinvar Road	Calgary	Alberta
456889123	Flanders	Ned	60 Evergreen Terrace	Springfield	Alberta
413754621	Kennedy	Leon	808, 4900 Wildman Ave	Racoon City	Alberta
456438624	Lemoy	Leonard	55 Logic Way	Vulcan	Alberta
666666667	Mason	Harry	7 Luckstone Dr	Silent Hill	Alberta
666666666	Morris	Heather	7 Luckstone Dr	Silent Hill	Alberta
444638047	Redfield	Claire	653 Wildpark Place	Racoon City	Alberta
123115323	Simcox	Cole	311 Ocean View Drive	Vancouver	British C
456789124	Simpson	Homer	59 Evergreen Terrace	Springfield	Alberta
123456789	Smith	John	123 Peanut Lane	Calgary	Alberta
666666668	Sunderland	James	7 Heartbroken Ave	Silent Hill	Alberta
620451097	Williams	Amanda	25 Rodeo Drive	Edmonton	Alberta
666666669	Wolf	Claudia	66 Twisted View	Silent Hill	Alberta
371988812	Carswell	Mary	425 Remington Ave	Calgary	Alberta

One record, ‘Simpson, Homer’

Storing Information In A Database (3)

- Field: are attributes used to describe each record in a table



Fields of the table

SIN	LastName	FirstName	Address	City	Province
638666670	Cartland	Douglas	1109, 4944 Dalworth Dr	Silent Hill	Alberta
456789123	Cartman	Eric	456 Lynchview Road	Southpark	Alberta
670380456	Edgar	Maureen	300, Lockinvar Road	Calgary	Alberta
456889123	Flanders	Ned	60 Evergreen Terrace	Springfield	Alberta
413754621	Kennedy	Leon	808, 4900 Wildman Ave	Racoon City	Alberta
456438624	Lemoy	Leonard	55 Logic Way	Vulcan	Alberta
666666667	Mason	Harry	7 Luckstone Dr	Silent Hill	Alberta
666666666	Morris	Heather	7 Luckstone Dr	Silent Hill	Alberta
444638047	Redfield	Claire	653 Wildpark Place	Racoon City	Alberta
123115323	Simcox	Cole	311 Ocean View Drive	Vancouver	British C
456789124	Simpson	Homer	59 Evergreen Terrace	Springfield	Alberta
123456789	Smith	John	123 Peanut Lane	Calgary	Alberta
666666668	Sunderland	James	7 Heartbroken Ave	Silent Hill	Alberta
620451097	Williams	Amanda	25 Rodeo Drive	Edmonton	Alberta
666666669	Wolf	Claudia	66 Twisted View	Silent Hill	Alberta
371988812	Carswell	Mary	425 Remington Ave	Calgary	Alberta

'Address' field describes location

Tables Used In The Example

- **This example can be found online:**
  - http://pages.cpsc.ucalgary.ca/~tamj/203/notes/data/access_example.mdb

- **Employees table (tracks information about individual employees)**
  - SIN
  - LastName
  - FirstName
  - Address
  - City
  - Province
  - PostalCode
  - HomePhone
  - BirthDate
  - PayRate

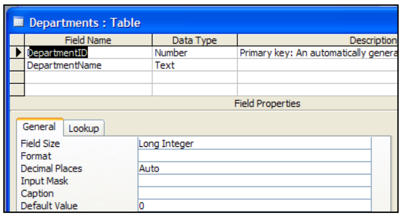
Tables Used In The Example (2)

- **Departments table (maps each department to a number e.g., Human Resources = 1, Marketing = 2 etc.)**
  - DepartmentID
  - DepartmentName

- **TimeBilled table (for each pay period information about how many hours each employee worked and how much they are owed is tracked with this table).**
  - TimeBilledID
  - EmployeeID
  - DepartmentID
  - StartPayPeriod
  - HoursWorked

MS-Access: Views Of Your Database

- **Design view**

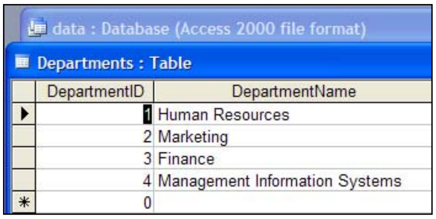


Field Name	Data Type	Description
DepartmentID	Number	Primary key: An automatically genera
DepartmentName	Text	

Field Size	Long Integer
Format	
Decimal Places	Auto
Input Mask	
Caption	
Default Value	0

  - Typically start with this view
  - Used to specify what fields that a table will consist of:
    - e.g., DepartmentID, DepartmentName
  - Used to specify the type and the format of the information in each field:
    - e.g., SIN is field with 9 characters that must be in the format 000 000 000

- **Datasheet view**



DepartmentID	DepartmentName
1	Human Resources
2	Marketing
3	Finance
4	Management Information Systems
0	

  - Once the fields have been specified in the Design view using the Datasheet view allows each record to be entered.

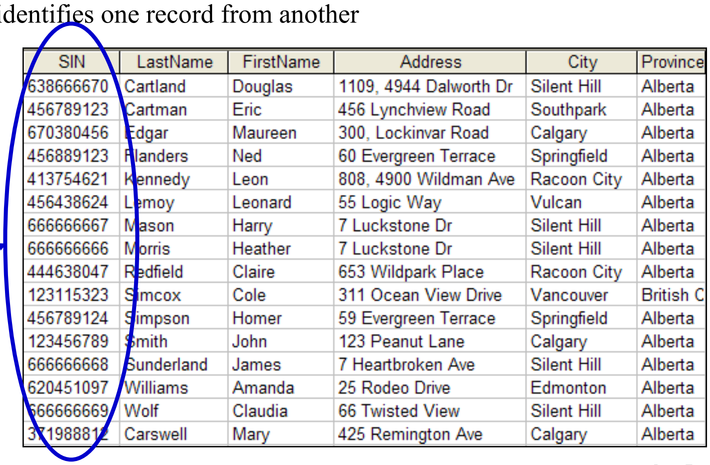
Types Of Tables

- **Data tables**
  - Stores data that provides information about the database
  - Dynamic, will likely be manipulated over the life the database (add, delete, modify)
  - E.g. Employees, TimeBilled tables (address and hours worked may change over time)
- **Validation tables**
  - Used to ensure data integrity (to ‘lookup’ values)
  - Typically it maps one value to another (e.g., product to product code, book to ISBN number)
  - Rarely (if ever) changes
  - E.g., Departments table

DepartmentID	DepartmentName
1	Human Resources
2	Marketing
3	Finance
4	Management Information Systems

Primary Key

- Each table should typically have one field designated as the primary key:
  - The primary key must be unique
  - It uniquely identifies one record from another



SIN	LastName	FirstName	Address	City	Province
638666670	Cartland	Douglas	1109, 4944 Dalworth Dr	Silent Hill	Alberta
456789123	Cartman	Eric	456 Lynchview Road	Southpark	Alberta
670380456	Edgar	Maureen	300, Lockinvar Road	Calgary	Alberta
456889123	Flanders	Ned	60 Evergreen Terrace	Springfield	Alberta
413754621	Kennedy	Leon	808, 4900 Wildman Ave	Racoon City	Alberta
456438624	Lemoy	Leonard	55 Logic Way	Vulcan	Alberta
666666667	Mason	Harry	7 Luckstone Dr	Silent Hill	Alberta
666666666	Morris	Heather	7 Luckstone Dr	Silent Hill	Alberta
444638047	Redfield	Claire	653 Wildpark Place	Racoon City	Alberta
123115323	Simcox	Cole	311 Ocean View Drive	Vancouver	British C
456789124	Simpson	Homer	59 Evergreen Terrace	Springfield	Alberta
123456789	Smith	John	123 Peanut Lane	Calgary	Alberta
666666668	Sunderland	James	7 Heartbroken Ave	Silent Hill	Alberta
620451097	Williams	Amanda	25 Rodeo Drive	Edmonton	Alberta
666666669	Wolf	Claudia	66 Twisted View	Silent Hill	Alberta
371988812	Carswell	Mary	425 Remington Ave	Calgary	Alberta

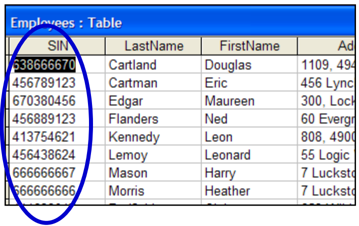
Primary Key for table ‘Employees’ is the ‘SIN’ field

Choosing A Primary Key

- A primary key must be unique to each record because it is the one thing that distinguishes them.
- If there is at least (or even exactly) one instance (however unlikely) where records can take on the same value for a field then that field cannot be a primary key. (When in doubt if this will ever be the case then verify with your users).
- If a single key field cannot be found then several fields can be combined into a composite key. (Each field is still a separate field but together they form a unique primary key for each record).
- If a unique primary key still cannot be found then 'invent' one.

Foreign Key

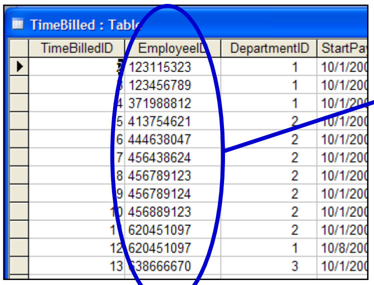
- A key in one table that refers to a key in another field:
  - E.g. for example: EmployeeID for the TimeBilled table and SIN for the Employees table.



Employees : Table

SIN	LastName	FirstName	Ad
638666670	Cartland	Douglas	1109, 494
456789123	Cartman	Eric	456 Lync
670380456	Edgar	Maureen	300, Lock
456889123	Flanders	Ned	60 Evergr
413754621	Kennedy	Leon	808, 4900
456438624	Lemoy	Leonard	55 Logic
666666667	Mason	Harry	7 Luckst
666666666	Morris	Heather	7 Luckst

SIN: Primary key for 'Employees' table



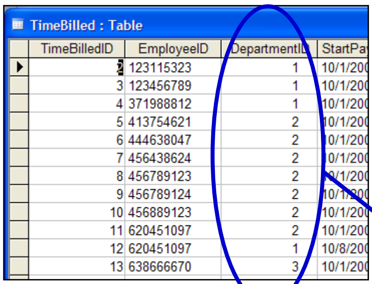
TimeBilled : Table

TimeBilledID	EmployeeID	DepartmentID	StartPa
2	123115323	1	10/1/200
3	123456789	1	10/1/200
4	371988812	1	10/1/200
5	413754621	2	10/1/200
6	444638047	2	10/1/200
7	456438624	2	10/1/200
8	456789123	2	10/1/200
9	456789124	2	10/1/200
10	456889123	2	10/1/200
11	620451097	2	10/1/200
12	620451097	1	10/8/200
13	638666670	3	10/1/200

EmployeeID is a foreign key of the 'TimeBilled' table that corresponds to the SIN primary key of the 'Employees' table

Foreign Key

- A key in one table that refers to a key in another field:
  - E.g. for example: DepartmentID field of the TimeBilled & Departments tables.

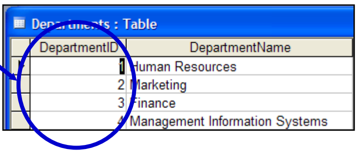


TimeBilled : Table

	TimeBilledID	EmployeeID	DepartmentID	StartPa
▶	2	123115323	1	10/1/200
	3	123456789	1	10/1/200
	4	371988812	1	10/1/200
	5	413754621	2	10/1/200
	6	444638047	2	10/1/200
	7	456438624	2	10/1/200
	8	456789123	2	10/1/200
	9	456789124	2	10/1/200
	10	456889123	2	10/1/200
	11	620451097	2	10/1/200
	12	620451097	1	10/8/200
	13	638666670	3	10/1/200

DepartmentID: A foreign key of the 'TimeBilled' table that corresponds to the primary key in the 'Departments' table

DepartmentID: The primary key of the 'Departments' table

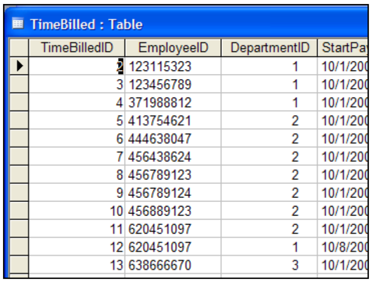


Departments : Table

	DepartmentID	DepartmentName
▶	1	Human Resources
	2	Marketing
	3	Finance
	4	Management Information Systems

Parent And Child Tables

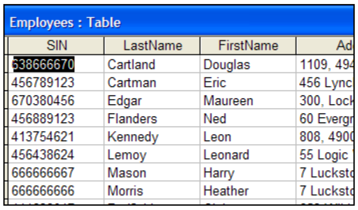
- Parent table: A table whose *primary key* is the foreign key of another table.
- Child table: The table whose *foreign key* is the primary key of another table.



TimeBilled : Table

	TimeBilledID	EmployeeID	DepartmentID	StartPa
▶	2	123115323	1	10/1/200
	3	123456789	1	10/1/200
	4	371988812	1	10/1/200
	5	413754621	2	10/1/200
	6	444638047	2	10/1/200
	7	456438624	2	10/1/200
	8	456789123	2	10/1/200
	9	456789124	2	10/1/200
	10	456889123	2	10/1/200
	11	620451097	2	10/1/200
	12	620451097	1	10/8/200
	13	638666670	3	10/1/200

EmployeeID is a foreign key of the 'TimeBilled' table that corresponds to the SIN primary key of the 'Employees' table (CHILD TABLE)



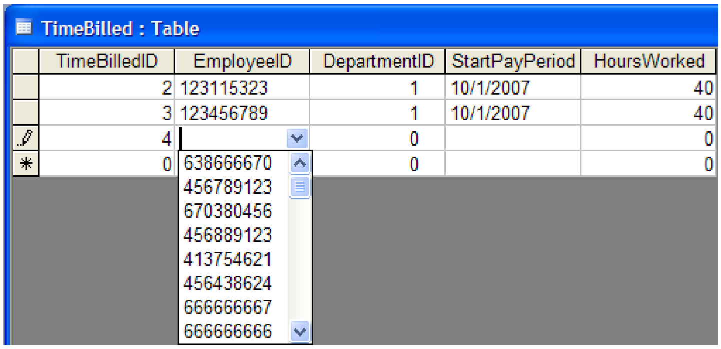
Employees : Table

SIN	LastName	FirstName	Ad
638666670	Cartland	Douglas	1109, 494
456789123	Cartman	Eric	456 Lync
670380456	Edgar	Maureen	300, Lock
456889123	Flanders	Ned	60 Evergr
413754621	Kennedy	Leon	808, 4900
456438624	Lemoy	Leonard	55 Logic
666666667	Mason	Harry	7 Luckst
666666666	Morris	Heather	7 Luckst

SIN: Primary key for 'Employees' table (PARENT TABLE)

Purpose Of Foreign Keys

- To ensure the integrity of the foreign key.
- (MS-Access: Ensure referential integrity): as new records are entered in a table with a foreign key as one of the fields, it will ensure that the record will only be entered with a foreign key value that is listed in the appropriate table.



TimeBilled : Table

	TimeBilledID	EmployeeID	DepartmentID	StartPayPeriod	HoursWorked
	2	123115323	1	10/1/2007	40
	3	123456789	1	10/1/2007	40
✎	4		0		0
*	0		0		0

638666670
456789123
670380456
456889123
413754621
456438624
666666667
666666666

Null Values

- Refers to empty fields of a record.
- Primary keys cannot be null but other fields may be null.

Types Of Data Integrity In Databases

1. Table-level integrity (entity integrity):
   - Ensuring that no duplicate records exist.
   - Ensuring that no primary keys are null: MS-Access (automatic) indexed – no duplicates.
2. Relationship-level integrity (referential integrity):
   - Ensuring that relationship between a pair of tables is sound and the records in the tables are synchronized when data is entered into, updated in or deleted from either table (MS-Access: only partially implemented).
3. Field-level integrity (domain integrity):
   - Ensuring that the values in each field are valid and accurate.
   - In MS-Access this is done through input masks and validation rules.

Input Masks

- Ensures the proper format for the data entered into the database
- From the example: SIN number in the Employees table must be entered as:
  - <**three digits**> <space> <**three digits**> <space> <**three digits**>
- Invalid inputs:
  - Abc def ghi
  - 321 22 4234

Validation Rules

- Validation rules check the data that is entered is in the correct range.
- From the example (all employ the logical AND):
  - ‘Employees’: BirthDate
  - ‘Employees’: PayRate
  - ‘TimeBilled’: HoursWorked

Guidelines For Naming Tables

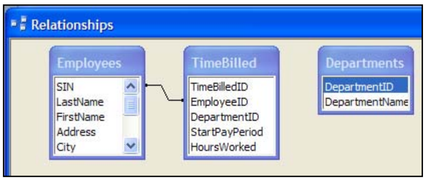
1. Create a unique and descriptive name.
2. Do not use words that convey physical characteristics or database terminology.
3. While names should be short avoid using acronyms and abbreviations unless they are well-known.
4. Do not use proper names or words that will restrict the type of data to be entered into the table.
5. Consider using the *plural* form of a name.
6. Avoid the use of spaces in names.

Guidelines For Naming Fields

1. Create a unique and descriptive name.
2. Create a name that accurately, clearly and unambiguously identifies the characteristic that the field represents.
3. While names should be short avoid using acronyms and abbreviations unless they are well-known.
4. Use the *singular* form of a name.
5. Avoid the use of spaces in names.

Relationships Between Tables

- Relationships occur when a field of one table is a foreign key in another table.



- Multiplicity: indicates how many instances of a particular item participate in the relationship:
  1. One to one
  2. One to many
  3. Many to many

Multiplicity

1. One to one relationships
   - One entity participates in the relationship from the ‘left’ and one entity participates in the relationship from the ‘right’.
   - Person : head
   - Worker : Social Insurance Number
   - This type of relationship is rare in databases
2. One to many relationships
   - On one side of the relationship one entity participates in the relationship while on the other side: zero or more entities may participate in the relationship.
   - Person : Hair
   - Employees : TimeBilled : Departments

Multiplicity (2)

3. Many to many relationships
   - On each side of the relationship zero or more entities may participate in the relationship.
   - Students : Classes

Multiplicity (3)

3. Many to many relationships
   - This type of relationship is not directly implemented in databases:

Students table

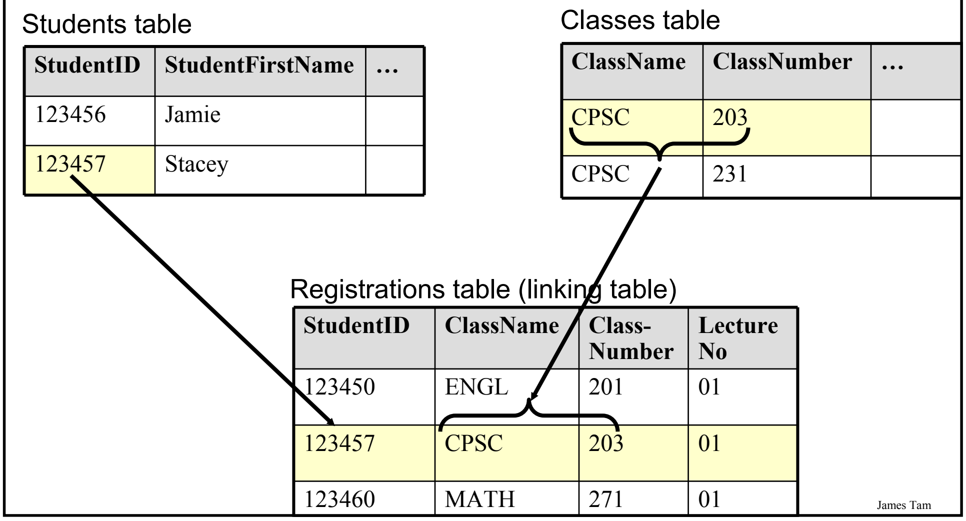
StudentID	StudentFirstName	StudentLastName	StudentPhone
123456	Jamie	Smyth	553-3992
123457	Stacey	Walls	790-3992
123458	Angel	Lam	551-4993

Classes table

ClassName	*ClassNumber*	*Lecture No*	ClassDescription
CPSC	*203*	*01*	Introduction to Computers
CPSC	*231*	*01*	Introduction to Computer Science I
CPSC	*233*	*01*	Introduction to Computer Science II

Multiplicity (4)

3. Many to many relationships
   - Typically implemented as two one to many relationships in databases:



Students table

StudentID	StudentFirstName	…
123456	Jamie	
123457	Stacey	

Classes table

ClassName	ClassNumber	…
CPSC	203	
CPSC	231	

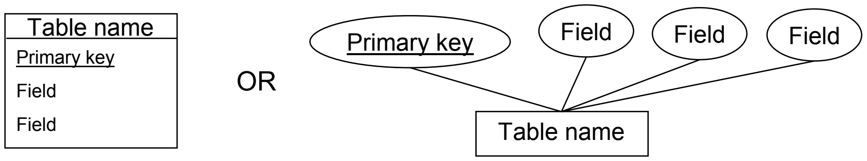
Registrations table (linking table)

StudentID	ClassName	Class-Number	Lecture No
123450	ENGL	201	01
123457	CPSC	203	01
123460	MATH	271	01

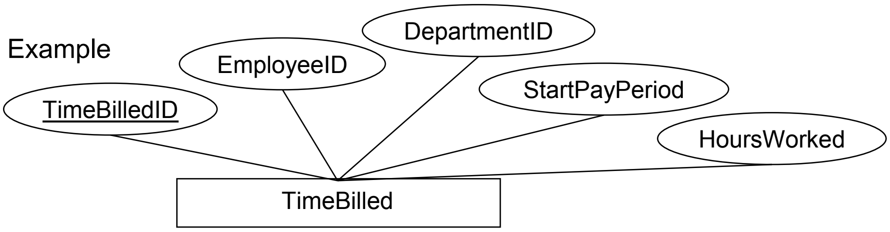
Diagrammatically Representing Databases

- Entity-Relation diagrams (E-R Diagrams or E.R.D.'s): show the fields of a table.

Format

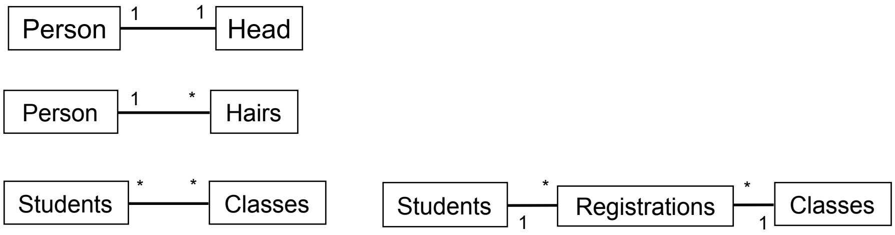


Example



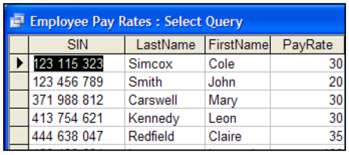
Diagrammatically Representing Relationships

- Graphically representing relationships between tables as well as any enforced rules on multiplicity:



Retrieving Data Via Queries

- Data retrieval occurs through the use of 'queries':
  - –A query is a question asked of the data in the database.
  - –Typically worded to show only the parts of the database for which the answer to the question is true.
  - –Example: What is the SIN, name and pay rate of every employee in the Employees Table:

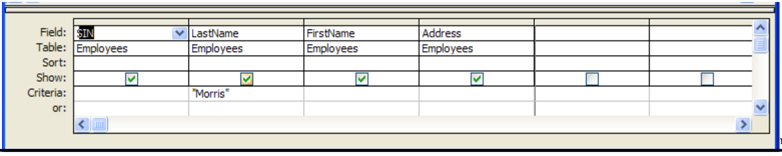


Employee Pay Rates : Select Query

SIN	LastName	FirstName	PayRate
123 115 323	Simcox	Cole	30
123 456 789	Smith	John	20
371 988 812	Carswell	Mary	30
413 754 621	Kennedy	Leon	30
444 638 047	Redfield	Claire	35

  - –Example: What employees have the last name of Morris?

Query

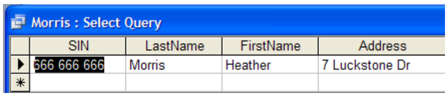


Field:	SIN	LastName	FirstName	Address		
Table:	Employees	Employees	Employees	Employees		
Sort:						
Show:	☑	☑	☑	☑	☐	☐
Criteria:		"Morris"				
or:						

Retrieving Data Via Queries (2)

- –Example: What employees have the last name of Morris?

Result of query

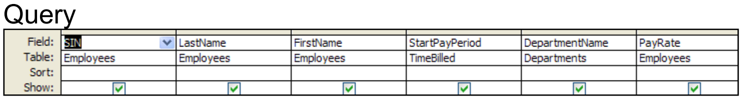


Morris : Select Query

SIN	LastName	FirstName	Address
666 666 666	Morris	Heather	7 Luckstone Dr

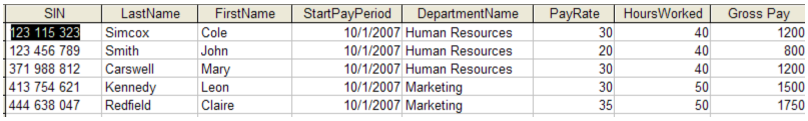
- Queries can search multiple tables:
  - –Example: What is the gross pay of employees (3 tables searched)?

Query



Field:	SIN	LastName	FirstName	StartPayPeriod	DepartmentName	PayRate
Table:	Employees	Employees	Employees	TimeBilled	Departments	Employees
Sort:						
Show:	☑	☑	☑	☑	☑	☑

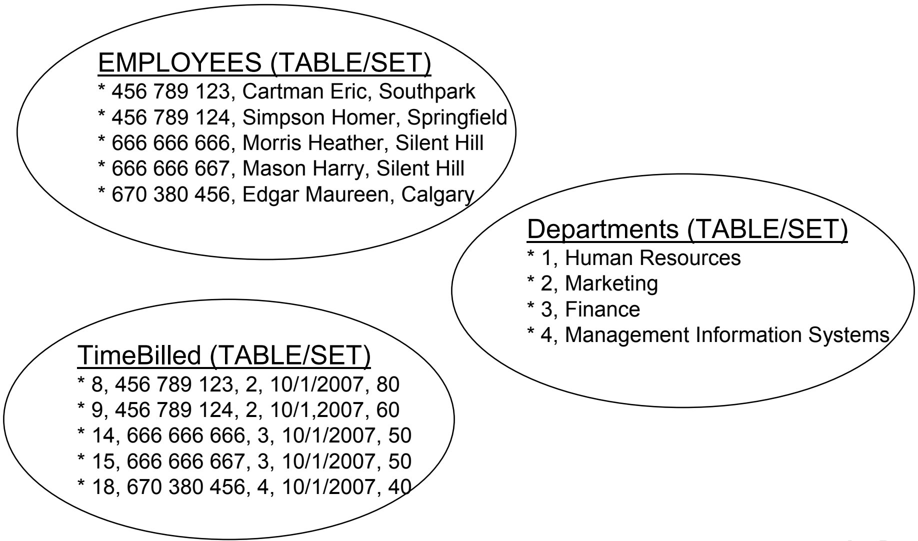
Result of the query



SIN	LastName	FirstName	StartPayPeriod	DepartmentName	PayRate	HoursWorked	Gross Pay
123 115 323	Simcox	Cole	10/1/2007	Human Resources	30	40	1200
123 456 789	Smith	John	10/1/2007	Human Resources	20	40	800
371 988 812	Carswell	Mary	10/1/2007	Human Resources	30	40	1200
413 754 621	Kennedy	Leon	10/1/2007	Marketing	30	50	1500
444 638 047	Redfield	Claire	10/1/2007	Marketing	35	50	1750

Databases And Set Theory

- Each table can be viewed as a set of information.



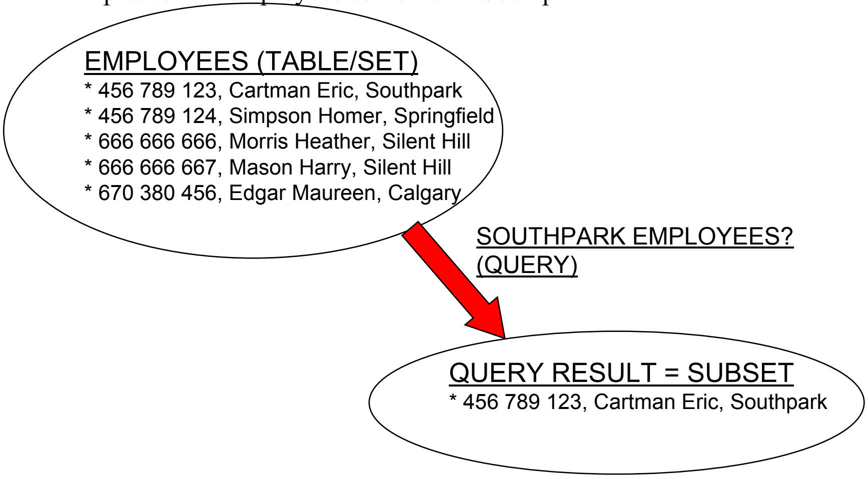
EMPLOYEES (TABLE/SET)
* 456 789 123, Cartman Eric, Southpark
* 456 789 124, Simpson Homer, Springfield
* 666 666 666, Morris Heather, Silent Hill
* 666 666 667, Mason Harry, Silent Hill
* 670 380 456, Edgar Maureen, Calgary

Departments (TABLE/SET)
* 1, Human Resources
* 2, Marketing
* 3, Finance
* 4, Management Information Systems

TimeBilled (TABLE/SET)
* 8, 456 789 123, 2, 10/1/2007, 80
* 9, 456 789 124, 2, 10/1,2007, 60
* 14, 666 666 666, 3, 10/1/2007, 50
* 15, 666 666 667, 3, 10/1/2007, 50
* 18, 670 380 456, 4, 10/1/2007, 40

Queries And Set Theory

- Queries retrieve a subset of the information:
  - Example: Which employees come from ‘Southpark’



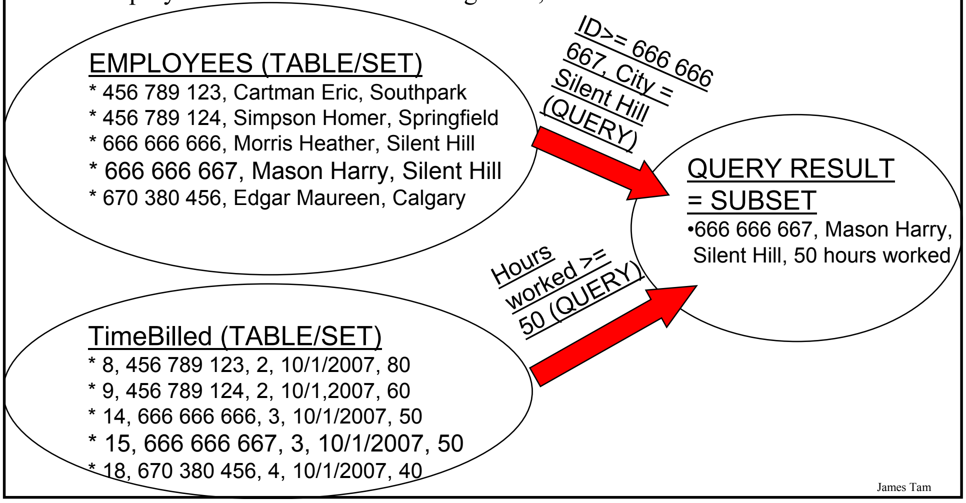
EMPLOYEES (TABLE/SET)
* 456 789 123, Cartman Eric, Southpark
* 456 789 124, Simpson Homer, Springfield
* 666 666 666, Morris Heather, Silent Hill
* 666 666 667, Mason Harry, Silent Hill
* 670 380 456, Edgar Maureen, Calgary

SOUTHPARK EMPLOYEES?
(QUERY)

QUERY RESULT = SUBSET
* 456 789 123, Cartman Eric, Southpark

Queries And Set Theory (2)

- Queries can be asked of multiple tables
  - Example: Which employees come from ‘Silent Hill’, and have an employee number 666 666 667 or greater, and worked 50 or more hours?



EMPLOYEES (TABLE/SET)

* 456 789 123, Cartman Eric, Southpark
* 456 789 124, Simpson Homer, Springfield
* 666 666 666, Morris Heather, Silent Hill
* 666 666 667, Mason Harry, Silent Hill
* 670 380 456, Edgar Maureen, Calgary

ID>= 666 666 667, City = Silent Hill (QUERY)

TimeBilled (TABLE/SET)

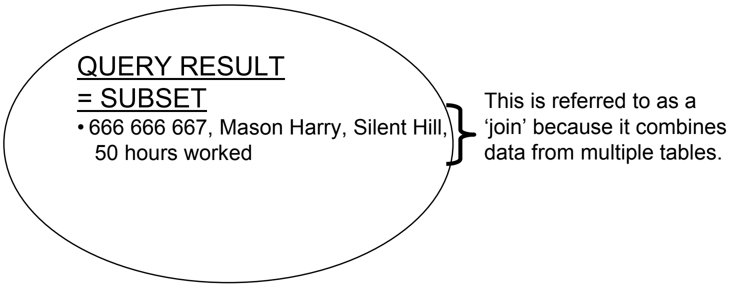
* 8, 456 789 123, 2, 10/1/2007, 80
* 9, 456 789 124, 2, 10/1,2007, 60
* 14, 666 666 666, 3, 10/1/2007, 50
* 15, 666 666 667, 3, 10/1/2007, 50
* 18, 670 380 456, 4, 10/1/2007, 40

Hours worked >= 50 (QUERY)

QUERY RESULT = SUBSET

•666 666 667, Mason Harry, Silent Hill, 50 hours worked

Queries And Set Theory (3)



QUERY RESULT = SUBSET

- 666 666 667, Mason Harry, Silent Hill, 50 hours worked

This is referred to as a ‘join’ because it combines data from multiple tables.

Logical Operations

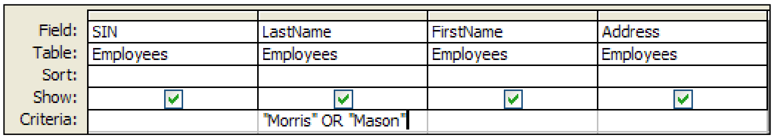
Operation	Description
AND	• All conditions must be true for the result to be true. • If any condition is false then the entire result is false.
OR	•All conditions must be false for the result to be false. •If any condition is true then the entire result is true.

Logical Comparisons

Operator	Description
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

Forming Queries

•Queries may be specified graphically:



Field:	SIN	LastName	FirstName	Address
Table:	Employees	Employees	Employees	Employees
Sort:				
Show:	☑	☑	☑	☑
Criteria:		"Morris" OR "Mason"		

•Also queries may be specified in the form of text descriptions of the question (SQL).

SQL (Structured Query Language)

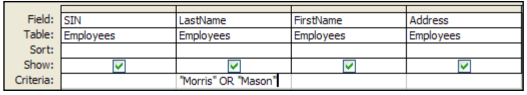
- It's the universal language for querying a relational database (very widely used!)
- The statements are portable between different database programs.
- Queries are formed using text descriptions (can be more powerful but more complex than graphical queries):
  - **SELECT**: Specifies the fields/columns shown in the query results e.g., SIN field.
  - **FROM**: Lists the tables from which the data is to be selected e.g., look in the Employees table.
  - **WHERE**: Provides the conditions to determine if rows/records are shown by the query.
  - **ORDER BY**: Specifies the order in which rows are to be returned by the query.

Note: Capitalizing of the words is a standard SQL convention.

Using Logic While Forming Queries

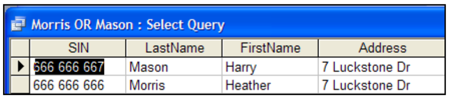
- Logical operators and logical comparisons can be performed during queries.
  - –Examples: Which employees have the last name of ‘Morris’ or ‘Mason’?

Query



Field:	SIN	LastName	FirstName	Address
Table:	Employees	Employees	Employees	Employees
Sort:				
Show:	☑	☑	☑	☑
Criteria:		"Morris" OR "Mason"		

Result of the query

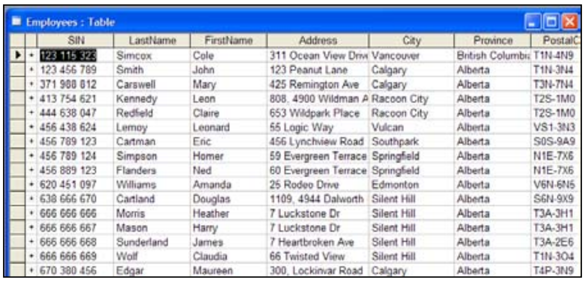


Morris OR Mason : Select Query

SIN	LastName	FirstName	Address
666 666 667	Mason	Harry	7 Luckstone Dr
666 666 666	Morris	Heather	7 Luckstone Dr

SQL Equivalent

- **(Employees table):**



Employees : Table

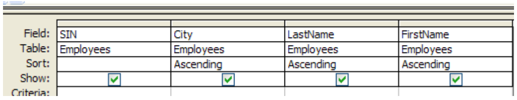
SIN	LastName	FirstName	Address	City	Province	PostalC
123 115 323	Simcox	Cole	311 Ocean View Driv	Vancouver	British Columbi	T1N-4N9
123 456 789	Smith	John	123 Peanut Lane	Calgary	Alberta	T1N-3N4
371 988 812	Carswell	Mary	425 Remington Ave	Calgary	Alberta	T3N-7N4
413 754 621	Kennedy	Leon	808, 4900 Wildman A	Racoon City	Alberta	T2S-1M0
444 638 047	Redfield	Claire	653 Wildpark Place	Racoon City	Alberta	T2S-1M0
456 438 624	Lemoy	Leonard	55 Logic Way	Vulcan	Alberta	VS1-3N3
456 789 123	Cartman	Eric	456 Lynchview Road	Southpark	Alberta	S0S-9A9
456 789 124	Simpson	Homer	59 Evergreen Terrace	Springfield	Alberta	N1E-7X6
456 889 123	Flanders	Ned	60 Evergreen Terrace	Springfield	Alberta	N1E-7X6
620 451 097	Williams	Amanda	25 Rodeo Drive	Edmonton	Alberta	V6N-6N5
638 666 670	Cartland	Douglas	1109, 4944 Dalworth	Silent Hill	Alberta	S6N-9X9
666 666 666	Morris	Heather	7 Luckstone Dr	Silent Hill	Alberta	T3A-3H1
666 666 667	Mason	Harry	7 Luckstone Dr	Silent Hill	Alberta	T3A-3H1
666 666 668	Sunderland	James	7 Heartbroken Ave	Silent Hill	Alberta	T3A-2E6
666 666 669	Wolf	Claudia	66 Twisted View	Silent Hill	Alberta	T1N-3O4
670 380 456	Edgar	Maureen	300, Lockinvar Road	Calgary	Alberta	T4P-3N9

- **SELECT** Employees.SIN, Employees.LastName, Employees.FirstName, Employees.Address
- **FROM** Employees
- **WHERE** (
- ((Employees.LastName)="Morris" Or (Employees.LastName)="Mason")
-)

Ordering Queries

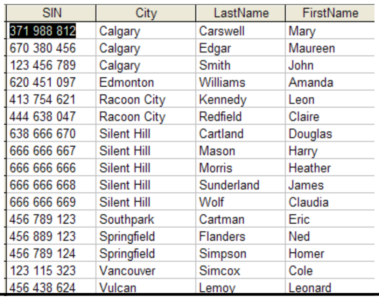
- Show the SIN, city, first name and last name of all employees in ascending order according to: city, last name and then first name.

Query



Field:	SIN	City	LastName	FirstName
Table:	Employees	Employees	Employees	Employees
Sort:		Ascending	Ascending	Ascending
Show:	☑	☑	☑	☑
Criteria:				

Query results



SIN	City	LastName	FirstName
371 988 812	Calgary	Carswell	Mary
670 380 456	Calgary	Edgar	Maureen
123 456 789	Calgary	Smith	John
620 451 097	Edmonton	Williams	Amanda
413 754 621	Racoon City	Kennedy	Leon
444 638 047	Racoon City	Redfield	Claire
638 666 670	Silent Hill	Cartland	Douglas
666 666 667	Silent Hill	Mason	Harry
666 666 666	Silent Hill	Morris	Heather
666 666 668	Silent Hill	Sunderland	James
666 666 669	Silent Hill	Wolf	Claudia
456 789 123	Southpark	Cartman	Eric
456 889 123	Springfield	Flanders	Ned
456 789 124	Springfield	Simpson	Homer
123 115 323	Vancouver	Simcox	Cole
456 438 624	Vulcan	Lemoy	Leonard

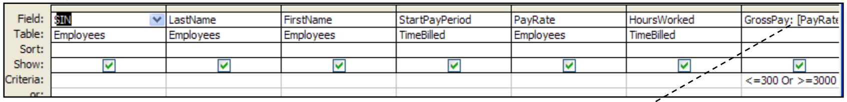
SQL Equivalent

- **SELECT** Employees.SIN, Employees.City, Employees.LastName, Employees.FirstName
- **FROM** Employees
- **ORDER BY** Employees.City, Employees.LastName, Employees.FirstName;

Queries With Ranges: Logical OR

- Ranges can be specified during the query.
  - –Example: Which employees have a gross pay on their time card that's less than $300 or greater than $3,000 (inclusive)?

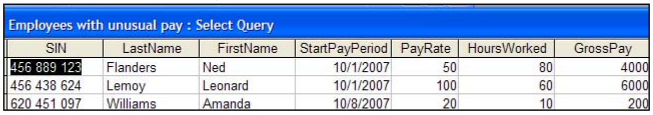
Query



Field:	SIN	LastName	FirstName	StartPayPeriod	PayRate	HoursWorked	GrossPay: [PayRate
Table:	Employees	Employees	Employees	TimeBilled	Employees	TimeBilled	
Sort:							
Show:	☑	☑	☑	☑	☑	☑	☑
Criteria:							<=300 Or >=3000
or:							

Calculated field
GrossPay: [PayRate]*[HoursWorked]

Result of the query



Employees with unusual pay : Select Query

SIN	LastName	FirstName	StartPayPeriod	PayRate	HoursWorked	GrossPay
456 889 123	Flanders	Ned	10/1/2007	50	80	4000
456 438 624	Lemoy	Leonard	10/1/2007	100	60	6000
620 451 097	Williams	Amanda	10/8/2007	20	10	200

SQL Equivalent

•**SELECT** Employees.SIN, Employees.LastName, Employees.FirstName, TimeBilled.StartPayPeriod, Employees.PayRate, TimeBilled.HoursWorked, [PayRate]*[HoursWorked] AS GrossPay

•**FROM** Employees JOIN TimeBilled ON Employees.SIN = TimeBilled.EmployeeID

•**WHERE** (

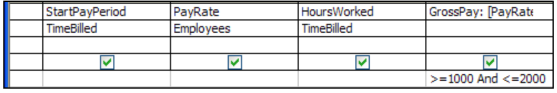
•(([PayRate]*[HoursWorked])<=300 Or ([PayRate]*[HoursWorked])>=3000))

•);

Queries With Ranges: Logical AND

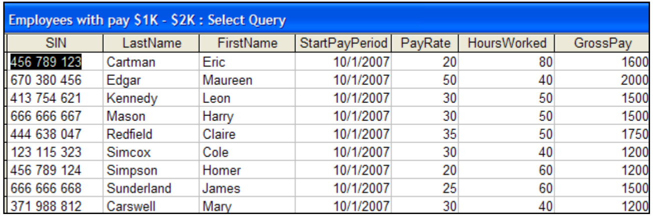
- Ranges can be specified during the query.
  - –Example: Which employees have a gross pay within the range of $1,000 - $2000 (inclusive) on one of their timecards?

Query



	StartPayPeriod	PayRate	HoursWorked	GrossPay: [PayRate
	TimeBilled	Employees	TimeBilled	
	☑	☑	☑	☑
				>=1000 And <=2000

Result of the query



Employees with pay $1K - $2K : Select Query

SIN	LastName	FirstName	StartPayPeriod	PayRate	HoursWorked	GrossPay
456 789 123	Cartman	Eric	10/1/2007	20	80	1600
670 380 456	Edgar	Maureen	10/1/2007	50	40	2000
413 754 621	Kennedy	Leon	10/1/2007	30	50	1500
666 666 667	Mason	Harry	10/1/2007	30	50	1500
444 638 047	Redfield	Claire	10/1/2007	35	50	1750
123 115 323	Simcox	Cole	10/1/2007	30	40	1200
456 789 124	Simpson	Homer	10/1/2007	20	60	1200
666 666 668	Sunderland	James	10/1/2007	25	60	1500
371 988 812	Carswell	Mary	10/1/2007	30	40	1200

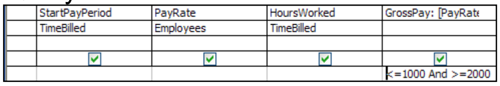
SQL Equivalent

- **SELECT** Employees.SIN, Employees.LastName, Employees.FirstName, TimeBilled.StartPayPeriod, Employees.PayRate, TimeBilled.HoursWorked, [PayRate]*[HoursWorked] AS GrossPay
- **FROM** Employees JOIN TimeBilled ON Employees.SIN = TimeBilled.EmployeeID
- **WHERE** (
- (([PayRate]*[HoursWorked])>=1000 And ([PayRate]*[HoursWorked])<=2000)
-);

Empty Queries

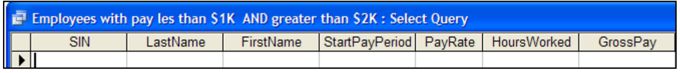
- Take care not to specify queries that can never be true!
- This will result in an “Empty Query”, a query that yields no results.
  - –Example: Which employees have a gross pay lower than $1,000 AND higher than $2,000 (inclusive for both) on one of their time cards?

Query



	StartPayPeriod	PayRate	HoursWorked	GrossPay: [PayRate
	TimeBilled	Employees	TimeBilled	
	☑	☑	☑	☑
				<=1000 And >=2000

Result of the (empty) query



Employees with pay les than $1K AND greater than $2K : Select Query

	SIN	LastName	FirstName	StartPayPeriod	PayRate	HoursWorked	GrossPay
▶							

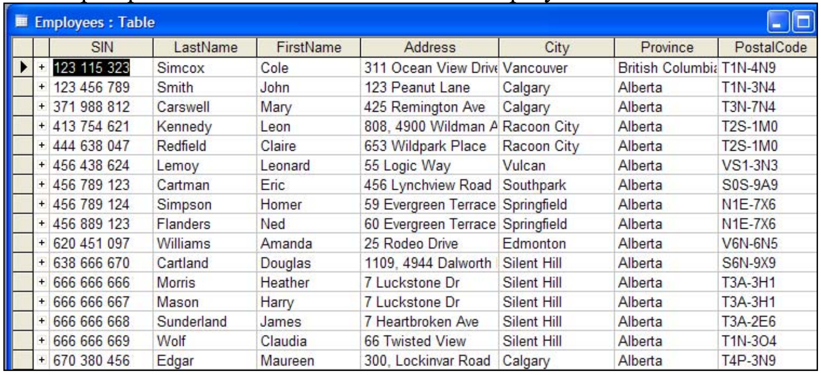
Wav file from “The Simpson” © Fox

SQL Equivalent

- **SELECT** TimeBilled.StartPayPeriod, Employees.PayRate, TimeBilled.HoursWorked, [PayRate]*[HoursWorked] AS GrossPay
- **FROM** Employees JOIN TimeBilled ON Employees.SIN = TimeBilled.EmployeeID
- **WHERE** (
- (([PayRate]*[HoursWorked])<=1000 And ([PayRate]*[HoursWorked])>=2000)
-);

Using The Wildcard In Queries

- The ‘wildcard’ character can stand for any number of characters in the position that it’s placed:
  - Example queries that follow will be in the Employees table:

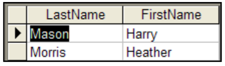


Employees : Table

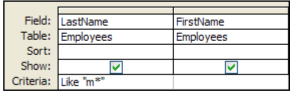
	SIN	LastName	FirstName	Address	City	Province	PostalCode
+	123 115 323	Simcox	Cole	311 Ocean View Drive	Vancouver	British Columbia	T1N-4N9
+	123 456 789	Smith	John	123 Peanut Lane	Calgary	Alberta	T1N-3N4
+	371 988 812	Carswell	Mary	425 Remington Ave	Calgary	Alberta	T3N-7N4
+	413 754 621	Kennedy	Leon	808, 4900 Wildman A	Racoon City	Alberta	T2S-1M0
+	444 638 047	Redfield	Claire	653 Wildpark Place	Racoon City	Alberta	T2S-1M0
+	456 438 624	Lemoy	Leonard	55 Logic Way	Vulcan	Alberta	VS1-3N3
+	456 789 123	Cartman	Eric	456 Lynchview Road	Southpark	Alberta	S0S-9A9
+	456 789 124	Simpson	Homer	59 Evergreen Terrace	Springfield	Alberta	N1E-7X6
+	456 889 123	Flanders	Ned	60 Evergreen Terrace	Springfield	Alberta	N1E-7X6
+	620 451 097	Williams	Amanda	25 Rodeo Drive	Edmonton	Alberta	V6N-6N5
+	638 666 670	Cartland	Douglas	1109, 4944 Dalworth	Silent Hill	Alberta	S6N-9X9
+	666 666 666	Morris	Heather	7 Luckstone Dr	Silent Hill	Alberta	T3A-3H1
+	666 666 667	Mason	Harry	7 Luckstone Dr	Silent Hill	Alberta	T3A-3H1
+	666 666 668	Sunderland	James	7 Heartbroken Ave	Silent Hill	Alberta	T3A-2E6
+	666 666 669	Wolf	Claudia	66 Twisted View	Silent Hill	Alberta	T1N-3O4
+	670 380 456	Edgar	Maureen	300, Lockinvar Road	Calgary	Alberta	T4P-3N9

Using The Wildcard In Queries (Access)

- Examples:
  - Which employees have a last name that begins with ‘m’?

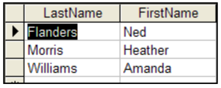


LastName	FirstName
Mason	Harry
Morris	Heather

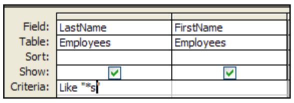


Field:	LastName	FirstName
Table:	Employees	Employees
Sort:		
Show:	☑	☑
Criteria:	Like "m*"	

  - Which employees have a last name ends with ‘s’?

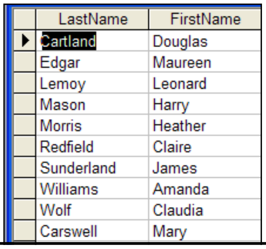


LastName	FirstName
Flanders	Ned
Morris	Heather
Williams	Amanda

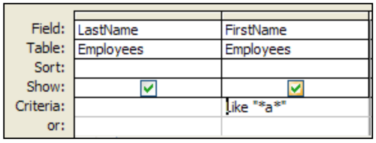


Field:	LastName	FirstName
Table:	Employees	Employees
Sort:		
Show:	☑	☑
Criteria:	Like "*s"	

  - Which employees have the letter ‘a’ anywhere in their first name?



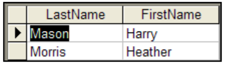
LastName	FirstName
Cartland	Douglas
Edgar	Maureen
Lemoy	Leonard
Mason	Harry
Morris	Heather
Redfield	Claire
Sunderland	James
Williams	Amanda
Wolf	Claudia
Carswell	Mary



Field:	LastName	FirstName
Table:	Employees	Employees
Sort:		
Show:	☑	☑
Criteria:		Like "*a*"
or:		

Using The Wildcard In Queries (SQL)

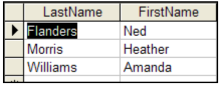
- Examples:
  - Which employees have a last name that begins with ‘m’?



LastName	FirstName
Mason	Harry
Morris	Heather

```
SELECT Employees.LastName,
Employees.FirstName
FROM Employees
WHERE (((Employees.LastName) Like "m*"));
```

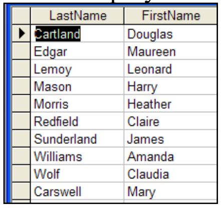
  - Which employees have a last name ends with ‘s’?



LastName	FirstName
Flanders	Ned
Morris	Heather
Williams	Amanda

```
SELECT Employees.LastName,
Employees.FirstName
FROM Employees
WHERE (((Employees.LastName) Like "*s"))
```

  - Which employees have the letter ‘a’ anywhere in their first name?

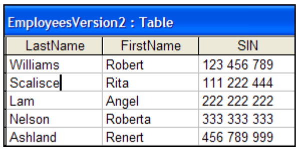


LastName	FirstName
Cartland	Douglas
Edgar	Maureen
Lemoy	Leonard
Mason	Harry
Morris	Heather
Redfield	Claire
Sunderland	James
Williams	Amanda
Wolf	Claudia
Carswell	Mary

```
SELECT Employees.LastName,
Employees.FirstName
FROM Employees
WHERE (((Employees.FirstName) Like "*a*"))
```

Single Character Wildcard

- The ‘?’ stands for a single character wildcard:
  - Querying the following table

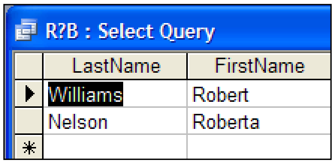


EmployeesVersion2 : Table

LastName	FirstName	SIN
Williams	Robert	123 456 789
Scalisce	Rita	111 222 444
Lam	Angel	222 222 222
Nelson	Roberta	333 333 333
Ashland	Renert	456 789 999

  - Which employees have the following string of characters in their first name: *<R> <any character> <B> <any number of characters>*

•



R?B : Select Query

LastName	FirstName
Williams	Robert
Nelson	Roberta

Database Design (And Redesign)

- The design-redesign process is referred to as “normalization”
- Each stage of redesign is referred to as a “form”:
  - Stage 1: First normal form
  - Stage 2: Second normal form
  - Stage 3: Third normal form
  - (For the purposes of this course getting a database into third normal form is sufficient although there are other stages as well).

Why Is Normalization Necessary?

- Normalization is regarded as good style
- It also helps to prevent errors or problems which are caused by the design of the database:
  - e.g., insertion anomalies: difficulties when adding new information
  - e.g., deletion anomalies: deleting information may result in the inadvertent loss of information

Example Database Table: Projects[1]

- This table shows:
  - ResearcherID: each professor working on a research project is given a computer generated login name.
  - Research project: name of the projects in a university department.
    - Professors can work on multiple projects
    - Research projects can be initiated without a professor
  - Location: room number of the research lab.

ResearcherID	Research projects	Location
aturing	Graphic Coloring Traveling Salesman	QC-103 QC-201
rdescartes	Knapsack	QC-121
cbabbage	Traveling Salesman Knapsack	QC-201 QC-121
bowen	Knapsack	QC-121

[1] From “Database Development for Dummies” by Allen G. Taylor

Problem: Some Cells Can Contain Multiple Entries

- Queries can be awkward to form
  - E.g., Using the ‘Like’ operator is difficult because it must deal with special cases (or more entries in each cell).

Databases In First Normal Form

- Each cell can contain *at most* one element (one value or a null value, the latter for non-primary key fields).
- The previous table in first normal form:

ResearcherID	Research project	Location
aturing	Graphic Coloring	QC-103
aturing	Traveling Salesman	QC-201
rdescartes	Knapsack	QC-121
cbabbage	Traveling Salesman	QC-201
cbabbage	Knapsack	QC-121
bowen	Knapsack	QC-121

First Normal Form: Critique

- **Improvements:**
  - Cells contain only one value which reduces some of the problems associated with forming queries.
- **Further improvements needed:**
  - There is redundancy in the table

ResearcherID	ResearchProject	Location
aturing	Graphic Coloring	QC-103
aturing	Traveling Salesman	QC-201

  - It may be subject to modification (addition and deletion) anomalies.

Deletion Anomaly

- Allan Turing ("aturing") no longer works on the "Graphic Coloring" project.

Before

ResearcherID	Research Project	Location
aturing	Graphic Coloring	QC-103
aturing	Traveling Salesman	QC-201
rdescartes	Knapsack	QC-121
cbabbage	Traveling Salesman	QC-201
cbabbage	Knapsack	QC-121
bowen	Knapsack	QC-121

After

ResearcherID	ResearchProject	Location
aturing	Traveling Salesman	QC-103
rdescartes	Knapsack	QC-121
cbabbage	Traveling Salesman	QC-201
cbabbage	Knapsack	QC-121
bowen	Knapsack	QC-121

Insertion Anomalies

- A new research project 'UFO' is added to the department and room 'Area-57' is to be used as the research lab but a researcher has not been hired.
- This is an incomplete record that cannot be properly added to the database yet.

ResearcherID	Research project	Location
aturing	Graphic Coloring	QC-103
aturing	Traveling Salesman	QC-201
rdescartes	Knapsack	QC-121
cbabbage	Traveling Salesman	QC-201
cbabbage	Knapsack	QC-121
bowen	Knapsack	QC-121

Problem With This Table

- The ‘Projects’ table combines two related but separate concepts:
  - Which research project is a particular researcher working on
  - What is the location of a particular project

ResearcherID	Research project	Location
aturing	Graphic Coloring	QC-103
aturing	Traveling Salesman	QC-201

- It’s a sign that a single unique key cannot be assigned
- By itself this isn’t necessarily a problem (i.e., ‘ResearcherID’ and ‘Research project’ can form a composite primary key).
- But the non-primary key element “Location” depends only on a part of the primary key (“Research project”) which can lead to anomalies.

Databases In Second Normal Form

- Every non-primary key element must be dependent on the primary key (and the entire primary key if the key is composite).
- The previous table split into two tables that are each in second normal form.

ResearchProject

ResearcherID	Project
aturing	Graph coloring
rdescartes	Knapsack
cbabbage	Traveling Salesman
bowen	Knapsack

ResearchLocation

Project	Location
Graph coloring	QC-103
Knapsack	QC-121
Traveling Salesman	QC-201

Critique Of Second Normal Form

- Dependencies can still exist that affects the database but in a slightly more subtle fashion.
- All non-key fields are dependent upon the primary key but some may be dependent in an indirect fashion.

Example[1]: "SalaryRange" Table

ResearcherID	AcademicRank	RangeCode
eschroedinger	Full professor	4
pdirac	Associate professor	3
wheisenberg	Full professor	4
hbethe	Assistant professor	2
jwheeler	Adjunct professor	1

Non-key fields whose values are dependent on the primary key (second normal form)

Primary key

[1] From "Database Development for Dummies" by Allen G. Taylor

The Example In 2nd Normal Form Are Still Subject To Some Anomalies

- Example Professor Dirac leaves the university.

Before

ResearcherID	AcademicRank	RangeCode
eschroedinger	Full professor	4
pdirac	Associate professor	3
wheisenberg	Full professor	4
hbethe	Assistant professor	2
jwheeler	Adjunct professor	1

After

ResearcherID	AcademicRank	RangeCode
eschroedinger	Full professor	4
wheisenberg	Full professor	4
hbethe	Assistant professor	2
jwheeler	Adjunct professor	1

Problem With The Database (2nd Normal Form)

- While both non-key elements are dependent upon the primary key, with “RangeCode” that dependency is indirect.

ResearcherID	AcademicRank	RangeCode
eschroedinger	Full professor	4
pdirac	Associate professor	3

- “RangeCode” is dependent upon “AcademicRank” which is in turn dependent upon “ResearcherID”.
- This is referred to as a transitive dependency:

RangeCode → AcademicRank → ResearcherID

Third Normal Form

- A database in third normal form is in second normal form and has no transitive dependencies.
- Previous example in third normal form:

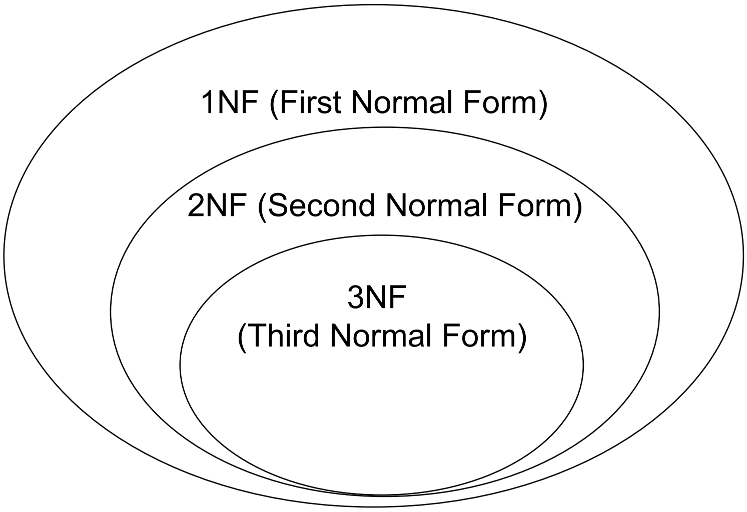
ResearcherRank

ResearcherID	AcademicRank
eschroedinger	Full professor
pdirac	Associate professor
wheisenberg	Full professor
hbethe	Assistant professor
jwheeler	Adjunct professor

RankRange

AcademicRank	RangeCode
Full professor	4
Associate professor	3
Assistant professor	2
Adjunct professor	1

The Normal Forms Have A Nested Structure



You Should Now Know

- How a database is broken down into tables and how tables are broken down into it's component parts
- What are the type of tables and the purpose of each
- What is the purpose of a primary key
- Principles for picking a good primary key
- What is a foreign key
- What is the purpose of creating a table with foreign keys
- What is a null value
- What are forms of data integrity exist in databases

You Should Now Know (2)

- Guidelines for naming tables and the fields of the tables
- What are the three relationships that may exist between tables and how they differ
- How is a many-to-many relationship typically implemented in a database
- The ERD representation of databases
- How to form different queries in order to retrieve data from a database (graphically and via SQL)
- What is an empty query
- How wildcards can be used in queries

You Should Now Know (3)

- How to normalize a database
- What are the characteristics of a database in: first normal form, second normal form, third normal form