MSc in Official Statistics
Statistical Computing:
Relational Databases

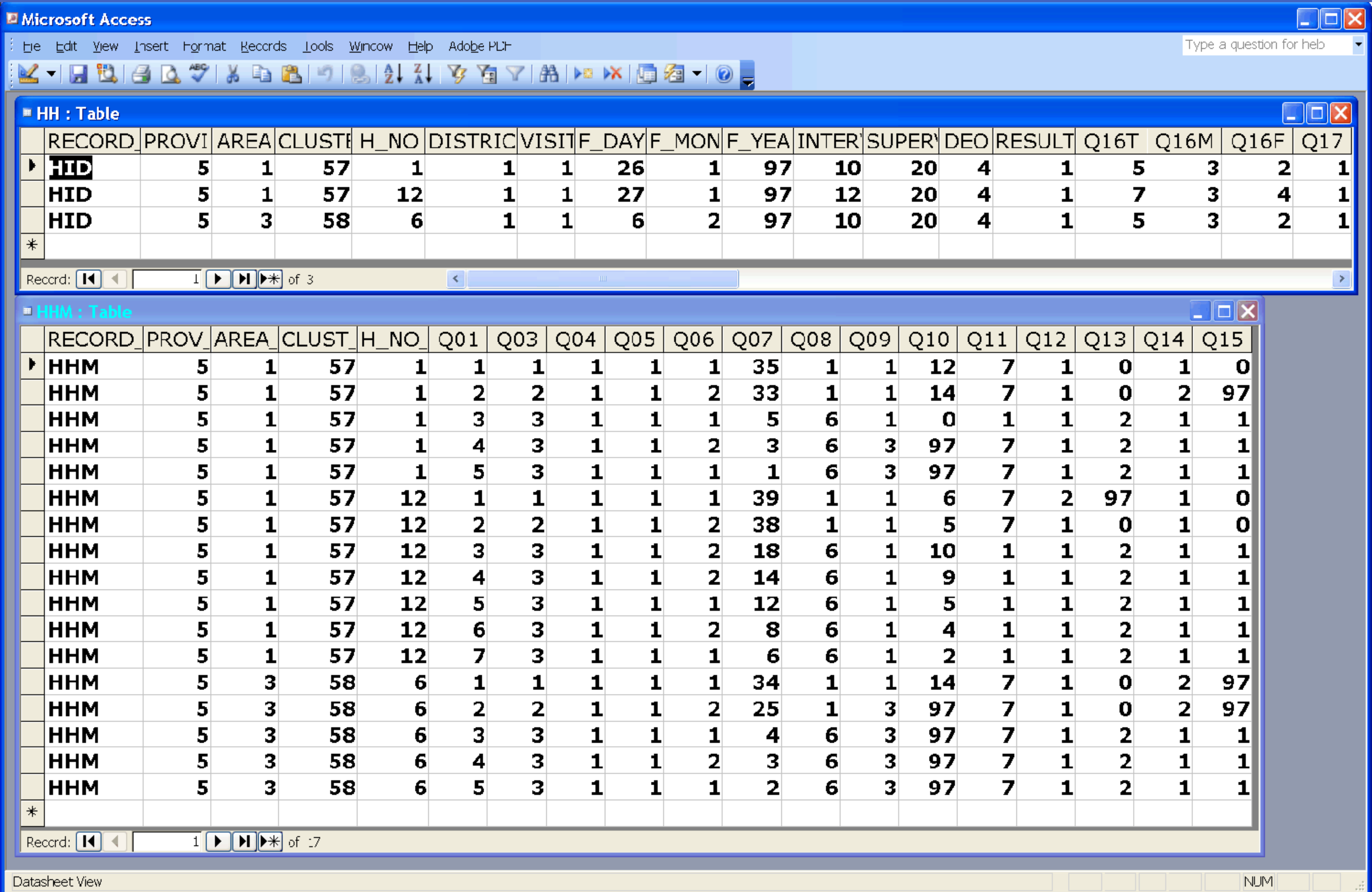
Andrew Westlake

Survey & Statistical Computing
63 Ridge Road, London N8 9NP, UK
+44 (0) 20 8374 4723
AJW@SaSC.co.uk (E-Mail)
www.SaSC.co.uk

The Relational Model

- A logical specification of the content and behaviour of a database management system, including
  - » The types of structure that can be present in a database
  - » The properties of elements that can be stored in these structures
  - » The operations that can be performed on these structures and their behaviour
  - » Facilities that must be present in the database management system
  - » The general nature of the interactions between the database and its users and administrators.

HH and HHM Sample Records



Microsoft Access
File Edit View Insert Format Records Tools Window Help Adobe PDF
Type a question for help

HH : Table

	RECORD_	PROVI	AREA	CLUSTE	H_NO	DISTRIC	VISIT	F_DAY	F_MON	F_YEA	INTER'	SUPER'	DEO	RESULT	Q16T	Q16M	Q16F	Q17
▸	HID	5	1	57	1	1	1	26	1	97	10	20	4	1	5	3	2	1
	HID	5	1	57	12	1	1	27	1	97	12	20	4	1	7	3	4	1
	HID	5	3	58	6	1	1	6	2	97	10	20	4	1	5	3	2	1
*																		

Record: |◀ ◀ 1 ▶ ▶| ▶* of 3

HHM : Table

	RECORD_	PROV_	AREA_	CLUST_	H_NO_	Q01	Q03	Q04	Q05	Q06	Q07	Q08	Q09	Q10	Q11	Q12	Q13	Q14	Q15
▸	HHM	5	1	57	1	1	1	1	1	1	35	1	1	12	7	1	0	1	0
	HHM	5	1	57	1	2	2	1	1	2	33	1	1	14	7	1	0	2	97
	HHM	5	1	57	1	3	3	1	1	1	5	6	1	0	1	1	2	1	1
	HHM	5	1	57	1	4	3	1	1	2	3	6	3	97	7	1	2	1	1
	HHM	5	1	57	1	5	3	1	1	1	1	6	3	97	7	1	2	1	1
	HHM	5	1	57	12	1	1	1	1	1	39	1	1	6	7	2	97	1	0
	HHM	5	1	57	12	2	2	1	1	2	38	1	1	5	7	1	0	1	0
	HHM	5	1	57	12	3	3	1	1	2	18	6	1	10	1	1	2	1	1
	HHM	5	1	57	12	4	3	1	1	2	14	6	1	9	1	1	2	1	1
	HHM	5	1	57	12	5	3	1	1	1	12	6	1	5	1	1	2	1	1
	HHM	5	1	57	12	6	3	1	1	2	8	6	1	4	1	1	2	1	1
	HHM	5	1	57	12	7	3	1	1	1	6	6	1	2	1	1	2	1	1
	HHM	5	3	58	6	1	1	1	1	1	34	1	1	14	7	1	0	2	97
	HHM	5	3	58	6	2	2	1	1	2	25	1	3	97	7	1	0	2	97
	HHM	5	3	58	6	3	3	1	1	1	4	6	3	97	7	1	2	1	1
	HHM	5	3	58	6	4	3	1	1	2	3	6	3	97	7	1	2	1	1
	HHM	5	3	58	6	5	3	1	1	1	2	6	3	97	7	1	2	1	1
*																			

Record: |◀ ◀ 1 ▶ ▶| ▶* of 17

Components of the Relational Model

- Domain
    - a set of possible (atomic) values
- Attribute
    - defined over a domain, has a name, cf. variable
- Tuple
    - a set of values, one associated with each attribute, cf. cases or records. NULL values supported
- Relation
    - defined over a set of attributes, has a name, consists of a set of tuples
- Relational DataBase
    - a set of relations

Relations

Relation	Cancer_ Registration						
Domain	Registration ID	Gender	Age in years	Weight in Kg, 1dp	SEG	Age in years	ICD
Attribute	ID	Sex	Age_at_ Registration	Weight	SEG	Age_at_ Diagnosis	Diagnosis
	4951	2	58	54.5	3	53	194
	4952	1	68	75.8	2	60	285
	4953	2	73	62.3	5	70	162
	4954	2	52	48.7	2	45	501
	4955	1	49	95.2	4	47	162
	4956	1	68	112.7	3	61	196
Tuples	4957	1	87	84.2	4	85	203
	4958	2	74	69.4	4	70	162
	4959	1	69	53.0	1	69	186
	4960	2	92	83.0	5	92	162
	4961	1	45	67.4	5	41	503

Relation	Lung_ Diseases	
Domain	ICD	String
Attribute	Diagnosis	Disease Name
	162	Cancer of the Lung
Tuples	501	Asbestosis
	503	Mesothelioma

- Rectangular Data tables, in which columns are variables, rows are cases.
- Tables can be linked by the values of common variables.

Components of a SQL database

- Data type
  - » Integer, Real, String, Date, Memo, etc
- Field
  - » defined over a data type, has a name, cf. variable. NULL values supported. Can have constraints
- Record
  - » a set of values, one associated with each field
- Table
  - » defined over a set of fields, has a name, consists of a set of records, can have keys and indexes
- SQL DataBase
  - » a set of tables, can have other properties, including relationships and implementation details
- PFFPS example

Data Definition in SQL

- Use Create statement to define structures

```
CREATE TABLE CANCER_REGISTRATION
  (ID   NUMBER(8) NOT NULL,
  SEX   INTEGER,
  AGE_AT_REGISTRATION NUMBER(3),
  WEIGHT      NUMBER(4,1)
  SEG   INTEGER,
  AGE_AT_DIAGNOSIS    NUMBER(3),
  DIAGNOSIS   NUMBER(3)
  Primary Key (ID) )
```

  » Does not provide proper Domains, only Data Types

  » Can include more constraints

- Design tools available in most systems - **Demo**

Data in a RDBMS Table

Table	Cancer_ Registration						
Data type	Number(8)	Integer	Number(3)	Number(4,1)	Integer	Number(3)	Number(3)
Column	ID	Sex	Age_at_ Registration	Weight	SEG	Age_at_ Diagnosis	Diagnosis
Rows	4951	2	58	54.5	3	53	194
	4952	1	68	75.8	2	60	285
	4953	2	73	62.3	5	70	162
	4954	2	52	48.7	2	45	501
	4955	1	49	95.2	4	47	162
	4956	1	68	112.7	3	61	196
	4957	1	87	84.2	4	85	203
	4958	2	74	69.4	4	70	162
	4959	1	69	53.0	1	69	186
	4960	2	92	83.0	5	92	162
	4961	1	45	67.4	5	41	503

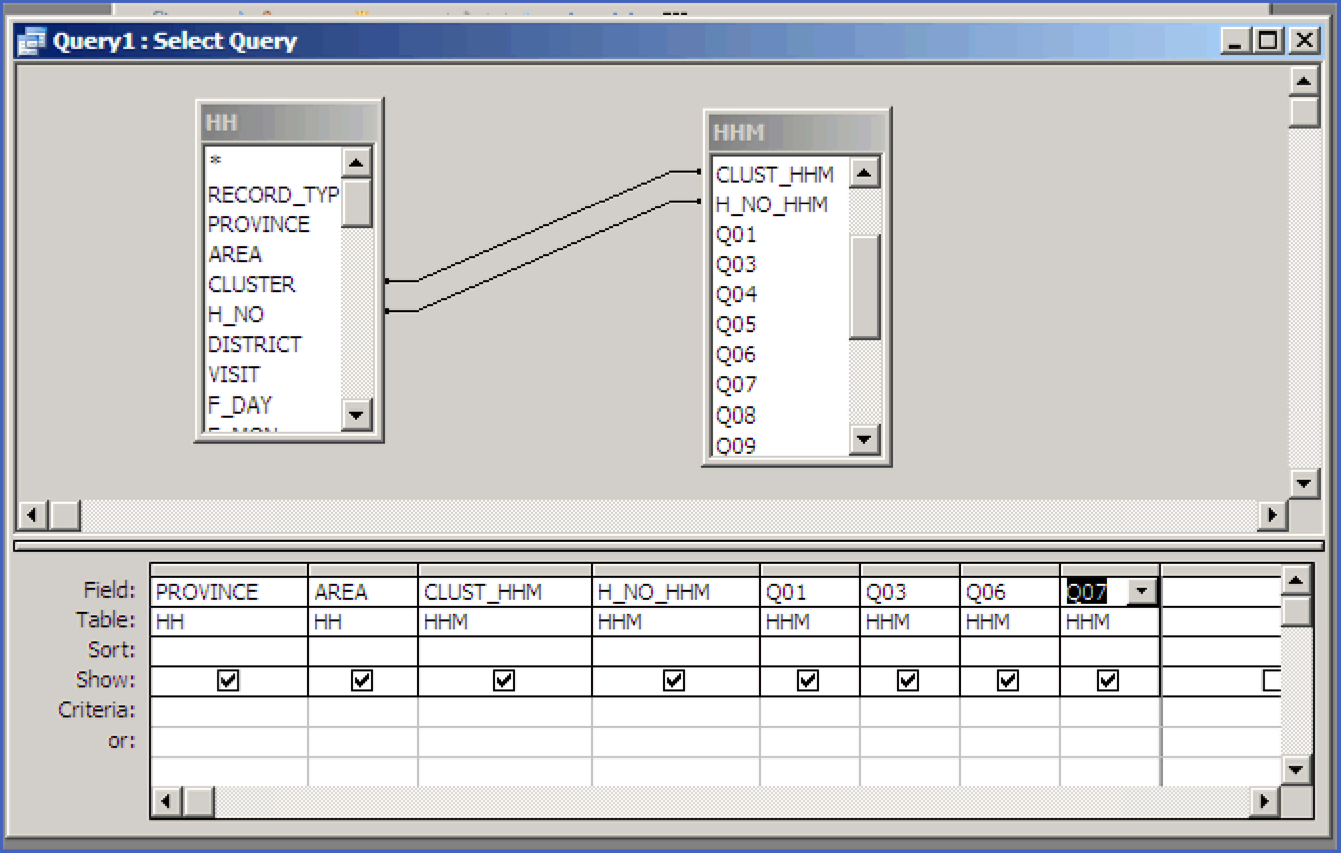
SQL Data Manipulation

- All manipulation (retrieval) of data uses the single Select statement, which has various components corresponding to different relational operations.
- The result of a SELECT statement is a relational table, which is displayed (by default) or can be stored or processed in another statement.
- Retrieval is usually done through a Query interface, which generates the SQL.

```
SELECT {DISTINCT}
    <expression list>
    FROM <table list>
    WHERE <condition>
    GROUP BY <column list>
    HAVING <group condition>
    ORDER BY <column list>
```

In Access, use the Query Definition interface to build Select statements - **Demo**

Query in MS Access



Field:	PROVINCE	AREA	CLUST_HHM	H_NO_HHM	Q01	Q03	Q06	Q07	
Table:	HH	HH	HHM	HHM	HHM	HHM	HHM	HHM	
Sort:									
Show:	☑	☑	☑	☑	☑	☑	☑	☑	□
Criteria:									
or:									

```
SELECT HH.PROVINCE, HH.AREA, HHM.CLUST_HHM,
HHM.H_NO_HHM, HHM.Q01, HHM.Q03, HHM.Q06, HHM.Q07
FROM HH INNER JOIN HHM ON (HH.H_NO = HHM.H_NO_HHM)
AND (HH.CLUSTER = HHM.CLUST_HHM);
```

Data Manipulation - Project and Restrict

R1			
A	B	C	D
1	b1	c1	d1
2	b2	c2	d2
3	b3	c3	d3

R2			
A	B	C	D
3	b3	c3	d3
4	b4	c4	d4
5	b5	c5	d5
6	b6	c6	d6
7	b7	c7	d7

- **Project** operation chooses columns

```
SELECT B, C FROM R1
SELECT   SEX, AGE_AT_DIAGNOSIS,
  DIAGNOSIS
  FROM   CANCER_REGISTRATION
```

- **Restrict** chooses rows (Where clause)

```
SELECT   * FROM R2 WHERE A < 6
SELECT   *
  FROM   CANCER_REGISTRATION
  WHERE  AGE_AT_DIAGNOSIS <= 65
```

Data Manipulation – Combining Tables

Union			
A	B	C	D
1	b1	c1	d1
2	b2	c2	d2
3	b3	c3	d3
4	b4	c4	d4
5	b5	c5	d5
6	b6	c6	d6
7	b7	c7	d7

- UNION operation combines rows from two tables. The number and type of variables must correspond.

```
SELECT * FROM R1
  UNION  SELECT * FROM R2
```

- Duplicate rows eliminated (by default, use `ALL` to retain)
- Statement is constructed from two Select statements.
- Difference and Intersection operations also available.

Data manipulation – Combining Columns

R3		
X	D	E
1	d3	e1
2	d5	e2
3	d3	e3

Product						
A	B	C	R1. D	X	R3. D	E
1	b1	c1	d1	1	d3	e1
1	b1	c1	d1	2	d5	e2
1	b1	c1	d1	3	d3	e3
2	b2	c2	d2	1	d3	e1
2	b2	c2	d2	2	d5	e2
2	b2	c2	d2	3	d3	e3
3	b3	c3	d3	1	d3	e1
3	b3	c3	d3	2	d5	e2
3	b3	c3	d3	3	d3	e3
4	b4	c4	d4	1	d3	e1
4	b4	c4	d4	2	d5	e2
4	b4	c4	d4	3	d3	e3

- The standard Cartesian **Product** operation is defined between any pair of relations.
  - » The result is a relation defined over the union of the sets of **attributes**, in which the tuples are the unions of each tuple from the first relation with every tuple from the second.

```
SELECT R1.*, R3.*  FROM  R1, R3
```

or

```
SELECT R1.*, R3.*  FROM  R1 CROSS JOIN R3
```

Data Manipulation - Join

- The **Join** operation is the combination of a **Product** operation with **Restrict** to select the rows of the result

```
SELECT R1.*, R3.*  FROM R1, R3  WHERE A = X or
SELECT R1.*, R3.*  FROM R1 INNER JOIN R3 ON A = X
```

- This is an **Equi-Join**
- **Natural Join** is based on columns with the same name

```
SELECT  R1.*, X, E  FROM  R1, R3
  WHERE R1.D = R3.D or
SELECT R1.*, R3.*  FROM R1 NATURAL JOIN R3
```

Join A=X

A	B	C	R1.D	X	R2.D	E
1	b1	c1	d1	1	d3	e1
2	b2	c2	d2	2	d5	e2
3	b3	c3	d3	3	d3	e3

Natural Join

A	B	C	D	X	E
3	b3	c3	d3	1	e1
3	b3	c3	d3	3	e3

Data Manipulation – Outer Join

- The **Outer Join** operation ensures that all the rows are included from one of the tables

```
SELECT R1.*, R3.*
   FROM R1 Left Join R3 ON R1.D = R3.D
```

Every row of the left table (R1) is included in the result, padded with Nulls if there was no match in the other table.

Outer Join						
A	B	C	R1.D	X	R3.D	E
1	b1	c1	d1	~	~	~
2	b2	c2	d2	~	~	~
3	b3	c3	d3	1	d3	e1
3	b3	c3	d3	3	d3	e3
4	b4	c4	d4	~	~	~

Data Manipulation – Conditional clauses

- Search predicates can take various forms, for example:
  - » comparisons, e.g.

    ```
    Age = 65
    Age < 45 and Age >= 15
    ```

  - » range predicates, e.g.

    ```
    Age BETWEEN 60 AND 75
    ```

  - » pattern-matches, e.g.

    ```
    Postcode LIKE 'LA8*'
    ```

  - » and list predicates, e.g.

    ```
    Diagnosis IN (162, 501, 503)
    ```

Data Manipulation – Sub-Queries

- Select statements that produce single values or lists for one field can be used within a Select statement anywhere where the corresponding value could be used.

```
SELECT     *      FROM   CANCER_REGISTRATION
  WHERE   DIAGNOSIS IN
  (SELECT DIAGNOSIS FROM LUNG_DISEASES)
```

  - » Note that the two references to Diagnosis are to different tables.

```
SELECT      COUNT(*) FROM CANCER_REGISTRATION
  WHERE     AGE_AT_REGISTRATION - AGE_AT_DIAGNOSIS  -  2 >
  (SELECT AVE( AGE_AT_REGISTRATION - AGE_AT_DIAGNOSIS )
   FROM     CANCER_REGISTRATION)
```

  - » This counts the number of registrations where the delay between diagnosis and registration is more than 2 years longer than the average such delay.

Data Manipulation – Computed Fields

- The expression list can contain expressions, so can be used to derive new variables for reporting or analysis (or to be stored).

```
SELECT     (AGE_AT_REGISTRATION - AGE_AT_DIAGNOSIS) AS DELAY
   FROM    CANCER_REGISTRATION
```

  - » Most things (including expressions) can be renamed by using a clause ‘AS name’.
  - » Arithmetic expressions can be based on fields, constants and sub-queries.
  - » Some systems allow function calls in expressions, but rules about the complexity allowed vary between systems. MS Access is very flexible when used with the Jet DB Engine.

Aggregation in Access

- Choose View/Totals
  - This invokes Grouping!
- Add the variables that define the grouping dimensions
  - Leave the Totals row as ‘Group by’
- Add the variables to be summarised
  - Change the Total row to the type of summary
  - For cell counts, choose any variable that does not contain Nulls
- The result is the correct (best) way to store aggregations within a relational database - **Demo**

Data Manipulation - Aggregation

- Aggregation functions can be used in the variable list

```
SELECT    COUNT(DIAGNOSIS), COUNT(DISTINCT DIAGNOSIS),
  AVE(WEIGHT)
  FROM    CANCER_REGISTRATION
```

  - » This produces one record for the whole table containing three values, the number of records with non-null values for Diagnosis, the number of distinct Diagnosis values, and the average Weight.
  - » The available functions are **COUNT, SUM, AVE, MIN, MAX**
  - » Aggregation functions can be applied to expressions.

Data Manipulation - Grouping

- The GROUP BY clause is used to perform aggregation within subgroups, producing a record for each group.

```
SELECT      DIAGNOSIS, COUNT(DIAGNOSIS), AVE(WEIGHT)
     FROM CANCER_REGISTRATION
     GROUP BY DIAGNOSIS
     ORDER BY DIAGNOSIS
```

  - » This produces a record for each diagnosis containing the number of registrations and their average weight as well as the diagnosis code.
  - » It is sorted into diagnosis code order, using the ORDER BY clause.
- Details
  - » The Group By clause can contain multiple fields (for cross-tabulation).
  - » The expression list can contain variables used in the Group By clause, and aggregate functions for any variable.
  - » A Where clause selects the records to be aggregated.
  - » A Having clause selects the groups to be returned.

Data Manipulation - Crosstabs

Mean Parity by Education, 5-year Age groups, PFFPS

Educ	Women	Wtd N	15	20	25	30	35	40	45
Above Secondary	585	905.89	0.00	0.31	1.18	2.43	3.03	3.94	2.44
None	8498	7119.23	0.17	1.21	2.88	4.70	5.78	6.23	7.19
Up to Middle	722	861.62	0.02	0.65	2.35	3.04	3.78	7.12	8.25
Up to Primary	1376	1456.77	0.10	0.87	2.37	4.23	4.12	6.18	5.75
Up to Secondary	844	954.42	0.01	0.39	1.48	3.09	4.24	4.71	5.24

- Special form of Aggregate Query (in Access) with one grouping factor converted to columns
  - » Can be stored as a table – this converts data values into Field names, but looses the classification structure
  - » Easy for simple cases (Access has a Wizard), more complex for tables as above, hard in SQL
  - » This table is not the correct relational structure for summary data

Data Manipulation – Crosstab SQL

```
TRANSFORM Sum(nz([q208])*[Std Weight])/Sum([Std Weight]) AS
   [Mean Parity]
SELECT IIf([Q09]<>1,"None",IIf([Q10]<6,"Up to
   Primary",IIf([q10]<9,"Up to Middle",IIf([Q10]>10,"Above
   Secondary","Up to Secondary")))) AS Educ,
   Count(PFFPSHHM.RECORD_TYP) AS Women, Sum([Std Weight]) AS
   [Wtd N]
FROM (PFFPSHHM INNER JOIN [Weight Std] ON CLUST_HHM =
   cluster) LEFT JOIN PFFPSWID ON (Q01 = L_NO_WID) AND
   (H_NO_HHM = H_NO_WID) AND (CLUST_HHM = CLUST_WID)
WHERE (((Q06)=2) AND ((Q07)>14 And (Q07)<50))
GROUP BY IIf([Q09]<>1,"None",IIf([Q10]<6,"Up to
   Primary",IIf([q10]<9,"Up to Middle",IIf([Q10]>10,"Above
   Secondary","Up to Secondary"))))
PIVOT Int([Q07]/5)*5;
```

Data Editing

- SQL has commands to insert, delete and change (update) rows in tables
- These commands are important at the programming level
- For direct manipulation of data these operations are usually done through Forms, or through special states of the Query generator

Views

- Stored query definition
  - » Important idea, with wide implications
- Result looks like a table
- Can be used like a table in many contexts
  - » Viewing data in the form needed by the user
    - Can sometimes use for data entry, but depends on the form of query
- Dynamic evaluation
  - » Ensures that the viewed information is up to date
    - May be inefficient if the information does not change

Keys

- A **Candidate Key** is a set of attributes which, taken together, uniquely identify each tuple.
    - » Several such Keys may exist, and at least one must always exist.
- The **Primary Key** for a relation is arbitrarily nominated from among these.
    - » The selection of a Key should be based on the conceptual uniqueness of the attributes (i.e. on the Domains), not on the actual (subset of possible) values in a relation at any particular time.
- A **Foreign Key** is defined over the same domain as a Primary Key, and so can provide a link between tuples.
    - » The Diagnosis column in the cancer registration table is a Foreign Key to the lung disease table.
    - » Household member identification includes the Household ID as a Foreign Key.
- Keys are usually implemented through **Indexes**
    - » An Index is a physical structure which stores information about the order and location of data values for a set of attributes, and which speeds up retrieval of subsets of records.
    - » Access defines Indexes (including the Primary Key) at table design, and Foreign Keys as Relationships.

Primary and Foreign Keys

Cancer_ Registration

Registration ID	Gender	Age in years	Weight in Kg, 1dp	SEG	Age in years	ICD
ID	Sex	Age_at_ Registration	Weight	SEG	Age_at_ Diagnosis	Diagnosis
4951	2	58	54.5	3	53	194
4952	1	68	75.8	2	60	285
4953	2	73	62.3	5	70	162
4954	2	52	48.7	2	45	501
4955	1	49	95.2	4	47	162
4956	1	68	112.7	3	61	196
4957	1	87	84.2	4	85	203
4958	2	74	69.4	4	70	162
4959	1	69	53.0	1	69	186
4960	2	92	83.0	5	92	162
4961	1	45	67.4	5	41	503
Primary Key						**Foreign Key**

Lung_ Diseases

ICD	String
Diagnosis	Disease_Name
162	Cancer of the Lung
501	Asbestosis
503	Mesothelioma
Primary Key	

Data Integrity – Codd's proposals

- Entity Integrity:
  - Every entity must have a proper existence, so no part of a primary key can be null.
    An attribute can be declared `NOT NULL` and `UNIQUE`.
    A `PRIMARY KEY` clause can be included, and a similar `UNIQUE` clause can identify sets of alternative candidate keys.
- Referential Integrity:
  - A foreign key must contain a value that is either all null, or equal to the primary key of some record in the referenced table.
    Implemented with the clause
    `FOREIGN KEY column list REFERENCES table`.
- Domain Integrity:
  - Every value for an attribute must be valid for the domain over which the attribute is defined. Not in SQL2.
- User-defined Integrity:
  - Rules about the consistency between attributes, depend on the underlying semantics of the application area.
    SQL2 includes the clause
    `CREATE ASSERTION name CHECK (condition)`
    to allow any condition to be evaluated on one or more columns in the table.

Current Implementations

- Stable, Mature products
  - » Major products easily scaleable across wide range of hardware. Oracle, MS SQL Server
  - » Good PC products now available, particularly Access, MySQL
- Useful Tool kits provided
  - » Data Entry and retrieval screens, report writers
  - » Active market in add-on products
- Client-Server facilities
  - » Many packages can act as clients, e.g. SAS, SPSS
  - » Efforts towards standardization of Client-Server communications, ODBC, ODAPI, XML
- Design tools
  - » Various systems for Entity-Relationship models, and accompanying code development

RDBMS Limitations

- Data Structures
  - » Aggregate (Macro) data, results of statistical processing
    - Can produce summary tables, but cannot associate additional semantics
    - OLAP facilities now integrated in some, eg MS SQL
- Data Types
  - » Dates and Times. Can store and manipulate, but no semantics.
  - » Complex structures – proposals in SQL:1999 et seq
    - Some systems now describe themselves as Object-Relational
- Meta Data
  - » Storage possible, but cannot add semantics
- Nulls
  - » Only one implemented
- Operation
  - » Optimized for commercial uses, transaction processing
  - » Designed for fixed problems, not dynamic data exploration
  - » Lack journal facility to record activities

Summary

- Relational databases are ubiquitous, and are useful for large-scale data collections
- Some manipulation and aggregation operations can be done more easily than in statistical packages
- Relational model is a useful way of thinking about data structures
- Implementations do not address issues of importance to Statisticians
- IT staff and Statisticians have different ways of thinking about data – we both have things to learn
- MS Access is a useful tool for manipulating moderate amounts of data with more complex structure
- No replacement for statistical packages for statistical analysis