



THE UNIVERSITY OF TEXAS
MD Anderson
Cancer Center
 Making Cancer History®

Databases in Radiotherapy and a brief introduction to SQL

- Peter Balter, Ph.D.

What is a database

- It is an organized collection of data
 - Could be a paper file system
 - Could be spread sheet
 (but neither are very good databases)



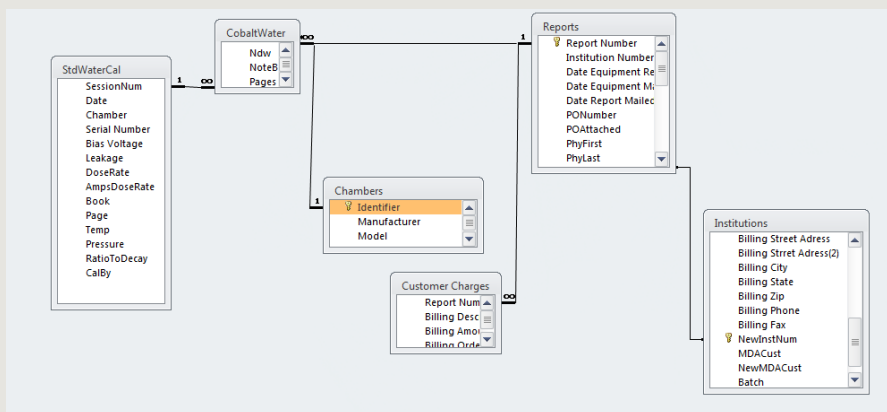
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	
1	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200

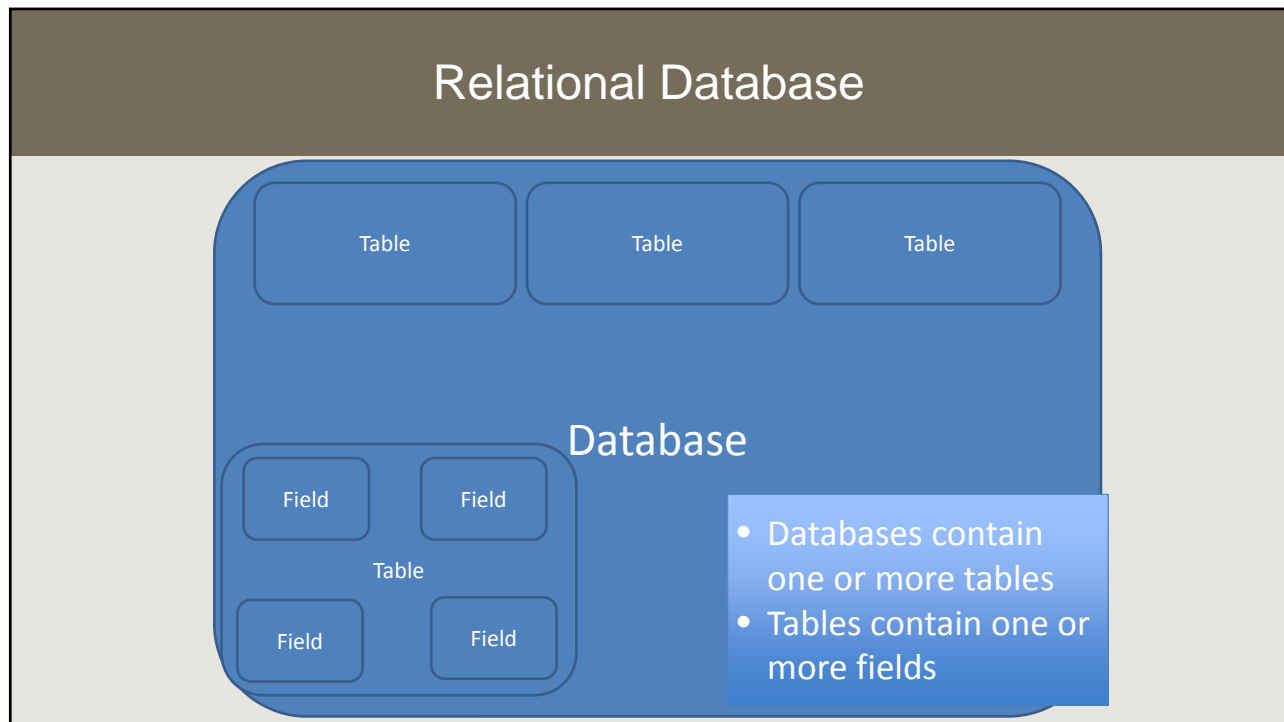
Databases in Radiotherapy

- Many of the systems in Radiotherapy are primarily databases with specialized equipment and user interfaces
 - MOSAIQ
 - Aria
 - Sun Nuclear Atlas
 - Raystation
- Other systems are not primarily database but generate a large amount of data that is organized using a database
 - Varian RPM
 - Radcalc
 - Pinnacle

Relational Database

- Data is grouped into tables based on its content
- The groupings have known relationships to each other
- Data access is done using these relationships





Post Relational Databases (noSQL)

- Considered by many to be a superior way to manage data
- Each record (document) carries its own fields and values with it (keys)
- Requires much less up-front work than a relational database
 - Does not require that relationships be defined between data elements
 - Allows modifications of database on-the-fly (no pre-defined schema)
- Showing up in many new systems
 - MOBIUS use MONGO and open source noSQL

Relational Database: Tables

- Each table generally relates to a real-world entity
 - Example: all contact information for an institution may be grouped in a single table

Field Name	Data Type
Institution Number	Number
PhName	Text
PhName	Text
PhTitle	Text
Name	Text
Department	Text
Street Address	Text
Street Address(2)	Text
City	Text
State	Text
Zip	Text
Phone Number	Text
Fax Number	Text
BillInst	Text
Billing Department	Text
Billing Street Address	Text
Billing Street Address(2)	Text
Billing City	Text
Billing State	Text
Billing Zip	Text
Billing Phone	Text
Billing Fax	Text
NewInstNum	Text
MDACust	Number
NewMDACust	Number

General (lookup)	
Field Size	50
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No

Relational Database: Fields

- Each record is stored in a table as a collection of fixed length fields
- Each field has a data type and a length
 - Similar to data type in programming languages
 - float, real, char, text, date
 - Each takes up a fixed amount of storage space
- In the example on the right ZIP code is stored as Text with 50 characters rather than numeric with 7 digits to allow free-form entry of foreign zip-codes (also worked well when +4 zips codes were introduced)
- Data validation can occur by database design or by rules enforced on entry

State	Text
Zip	Text
Phone Number	Text
Fax Number	Text
BillInst	Text
Billing Department	Text
Billing Street Address	Text
Billing Street Address(2)	Text
Billing City	Text
Billing State	Text
Billing Zip	Text
Billing Phone	Text
Billing Fax	Text
NewInstNum	Text
MDACust	Number
NewMDACust	Number

General (lookup)	
Field Size	50
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No
Indexed	No
Unicode Compression	Yes
IME Mode	No Control
IME Sentence Mode	None
Smart Tags	

Primary Keys

- Most tables have a primary key (field) that is used to define the relationship between that table with other tables
- The primary key can be the composite of 2 fields
- Often the primary key is a sequence number rather than a physical characteristic of the entity
 - Example: The primary key on a person could be their name but names change or could have been miss-spelled on entry so most databases with have a patient number usually independent of the hospital patient ID

PatientModel	
PK,FK5	PatientId
	PatientId2
	PatientFirstName
	PatientLastName
	PatientFullName
	PatientHonorific
	PatientDateOfBirth
	PatientAge
	PatientCitizenship
	PatientType
	PatientRace
	PatientSex
	PatientSSN
	PatientLanguage
	PatientAddressLine1
	PatientAddressLine2
	PatientCity
	PatientCountry

Example from Varian Unified Reports Application Schema

Foreign Keys

- Foreign Keys are fields in a table that point to the Primary Key of another table
 - Are restricted to being available in the other table to enforce data integrity

In this example the primary key for PatientInVivoDatModel is also a foreign key since inVivo dosimetry must be related to a unique patient

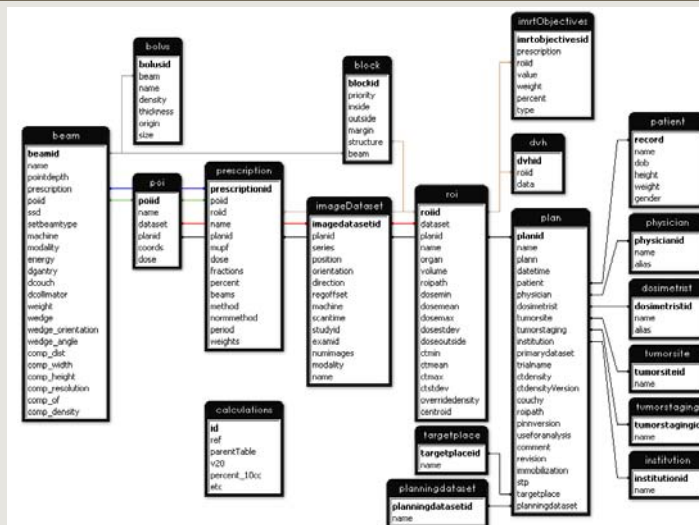
PatientInVivoDataModel	
PK,FK1	PatientId
FK4,FK5,FK6,FK7,FK8	ctrPatientSer
	ctrInVivoDosimetrySer
	InVivoDate
	InVivoVendorName
	DosimeterType
	DosimeterId
	DosimeterLocation
	FieldId
	FieldName
	CreatedDate

Example from Varian Unified Reports Application Schema

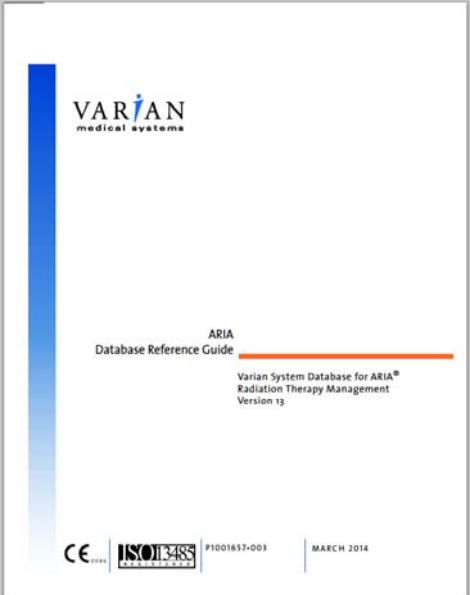
Database Schema/Data Dictionary

- Database Schema
 - The blueprint for the database
 - Shows how data is divided into tables and how tables relate to one another
 - Shows integrity constraints
 - Shows stored procedures
- Data Dictionary
 - Should include much of the same information as the schema
 - May contain further information
 - Detailed descriptions of the data

Typical Database Schema (in-house Pinnacle add-on database)



Data Dictionary(ARIA) - Public



Patient

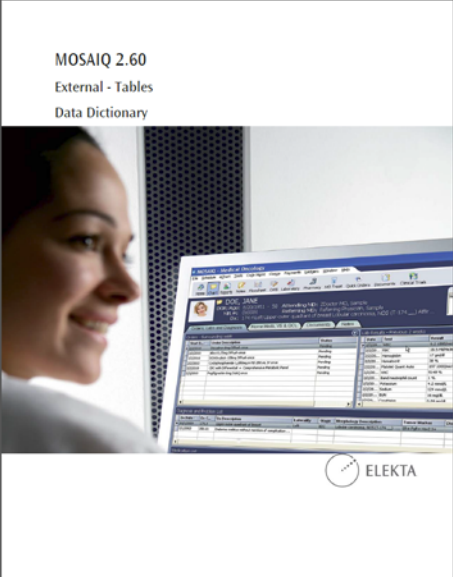
The *Patient* table stores information specific to a person who is a patient. This table provides demographic information about a patient. When a patient is deleted from the database, that person's demographic information remains in the database. Therefore the patient's ID cannot be reused. All other data is permanently removed from the database.


Column Name	Description	Datatype
PatientSer	System generated serial number for a person who is a patient.	VDI_SERIALNUMBER bigint
PatientId (AK1)	Unique patient identifier number assigned by the user when the information for this patient is entered.	VDI_PATIENTID nvarchar(25)
PatientId2 (EI)	Non-unique identifier that can be assigned to a patient.	VDI_PATIENTID nvarchar(25)
PatientUID (AK2)	Unique constraint Patient ID that will not be changed on upgrade, conversion, or deletion of the patient record.	VDI_UID nvarchar(94)
PatientType	Type of patient. Values are: <ul style="list-style-type: none"> • ActivePatient • ArchivedPatient • DeceasedPatient 	VDI_TABLENAME nvarchar(30)
CreationDate	Date and time of creation of this patient, or the date when this patient's data was imported.	VDI_DATETIMESTAMP Datetime
CreationUserName	User who created this patient record.	VDI_USERNAME nvarchar(32)
SSN	Patient's social security number.	VDI_STRING64 nvarchar(64)
FirstName	First name of patient.	VDI_NAME nvarchar(64)
MiddleName (EI)	Middle name of patient.	VDI_NAME nvarchar(64)

74 Database Reference Guide

Data Dictionary(MOSAIQ)- Available but not Public

MOSAIQ 2.60
External - Tables
Data Dictionary





IMPAC Data Dictionary Use and Non-Disclosure Agreement

To receive a data dictionary, you must complete the attached IMPAC Data Dictionary Use and Non-Disclosure Agreement and mail the original copy, signed in ink, to:

IMPAC Medical Systems
Attention: LEGAL AFFAIRS
100 Mathilda Place, 3rd Floor
Sunnyvale, California 94086-6076

Ensure that any information you provide in this document is legible. If it cannot be easily read, processing may be delayed.

Specify what software (MOSAIQ®, METRIC®, ANALYTIC®) and which version you are requesting.

The data dictionary is delivered through email. Please provide the email address to which the data dictionary should be sent.

If you have any questions, please feel free to contact Client Services for your region as indicated below.

North America: 855-683-5358 option 3, or support@impac.com
Europe and APAC: 00800 4000 5000 or +44 (0) 1293 654320, or support.europe@elektas.com
APAC: ANZ Software@elektas.com

* Faxed copies are not sufficient to procure a Data Dictionary.

IMPAC Medical Systems, Inc. • 100 Mathilda Pl. 3rd Floor Sunnyvale, CA 94086
• Phone: +1 (408) 639-8800 L16GK04M V01 1.0

Data Dictionary – other systems

- Most systems do not have publically available data-dictionaries
 - Few users are requesting them
- Most systems can be reasonably reverse engineered
 - Browse the data and table with an Database tool
 - Try and match data in tables to that in the GUI
- Many vendors will help with queries
 - The examples from Sun Nuclear Atlas were mostly hacked
 - Understanding how to re-derive data not stored directly in the database required help from the Vendor (which they provided)

Relationships:

- One-to-Many
 - A row in one table can have a large number of matching rows in another table
 - This is the most common relationship
 - Example: A single plan can have many beams but each beam can only belong to a single plan
- Many-to-Many
- One-to-One

Relationships:

- One-to-Many
- Many-to-Many
 - A row in one table can have a large number of matching rows in another table and vise-versa
 - Example: Plans can have multiple imagesets and imagesets can belong to multiple plans
- One-to-One

Relationships:

- One-to-Many
- Many-to-Many
- One-to-One
 - Each row in one table can have exactly one matching row in another table
 - Not common as if this relationship exists the rows could have been placed in the same table.

Queries and SQL

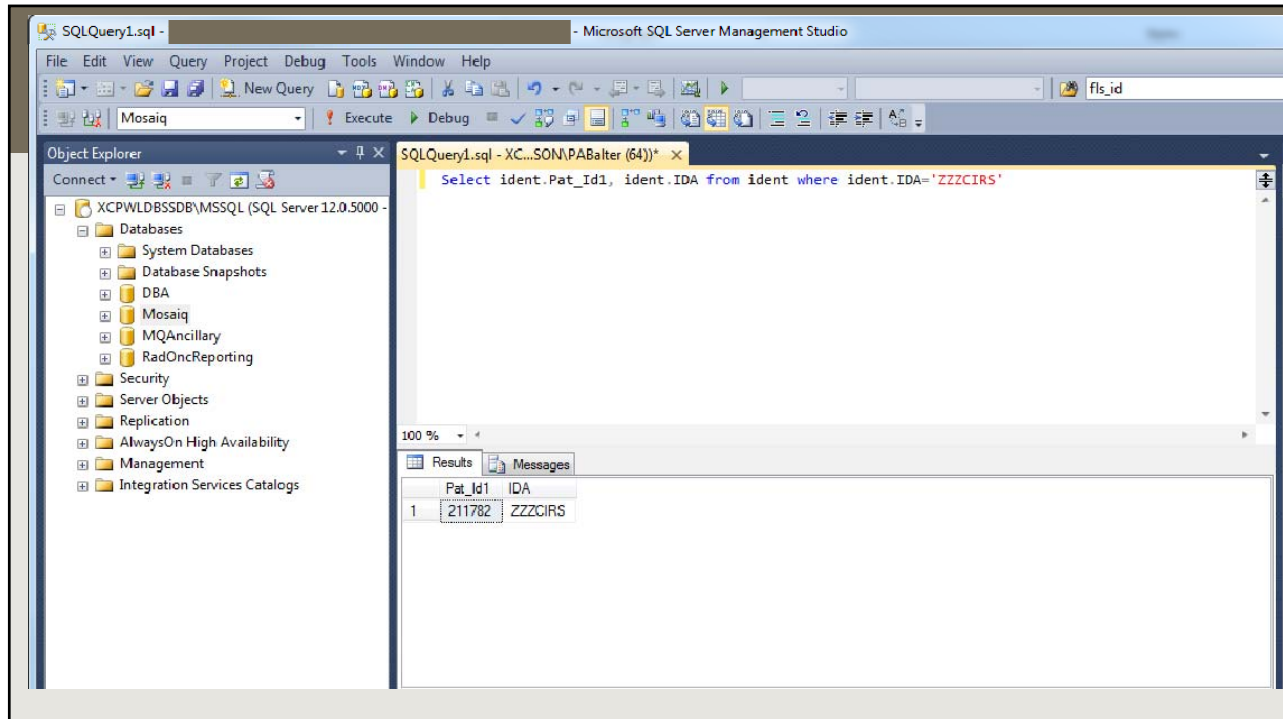
- SQL (SEQUEL): Structured English Query Language
- The standard means of retrieving data from a relational database
- Is very powerful but can be used simply
- The standard form is
 - Select <fields> from <table> where <criteria>;
- Queries can returned sorted results
 - Select <fields> from <table> where <criteria> ORDER BY <field>;

Simple Query Example MOSAIQ

Find patients internal ID based on their Hospital Medical Record Number

- Select ident.Pat_Id1, ident.IDA from ident where ident.IDA='ZZZCIRS'

Implemented in MicrosoftSQL Management Studio (but should work in any query engine that supports SQL)



Simple Query Example: Pinnacle

- Uses the built in PostgreSQL interactive terminal to find the number of patients in the active database
- `psql -d p3rtp -h localhost -U lpuser`
- Welcome to psql 8.3.5, the PostgreSQL interactive terminal.
- Type: `\copyright` for distribution terms
- `\h` for help with SQL commands
- `\?` for help with psql commands
- `\g` or terminate with semicolon to execute query
- `\q` to quit
- `p3rtp=> SELECT COUNT("patientid") FROM "patient";`
- `count`
- -----
- 1193
- (1 row)

Micheal Kantor

Example Query (Sun Nuclear Atlas)

- `SELECT dbo_Machine.MachineName FROM dbo_Machine`
returns an unsorted list of machines in the database
- `SELECT dbo_Machine.MachineName FROM dbo_Machine ORDER BY dbo_Machine.MachineName;`
returns a sorted list

MachineName	MachineName
2103	2100iX
2104	2103
2105	2104
2106	2105
2107	2106
2108	2107
2109	2108
2110	2109
Trilogy	2110
6EX	604
604	6EX
Primus 2	ACB1
ACB1	ACB2
ACB2	ACB3
ACB3	ACB4
ACB4	ACB5
ACB5	ACB6
ACB6	Aqusim1
Aqusim1	BAC eX
2100iX	BAC iX
TrueBeam	PET/CT
BAC iX	PNM eX
BAC eX	PNM iX

Queries and SQL using Joins

- Queries can span multiple tables using Joins
 - `Select <fields> from <table> Join <table2> on <common field> where <criteria>;`
 - These tables can span different databases
- Joins can be Inner or Outer Joins
 - Inner joins (default join) only returns data that has matching rows in both tables
 - Outer joins will give data that exists in either table
- Example if you have a table of equipment and one of calibrations and inner join on equipment will only give equipment with calibrations an out join will give all equipment if if no calibration exists

Simple Query Example MOSAIQ with Join

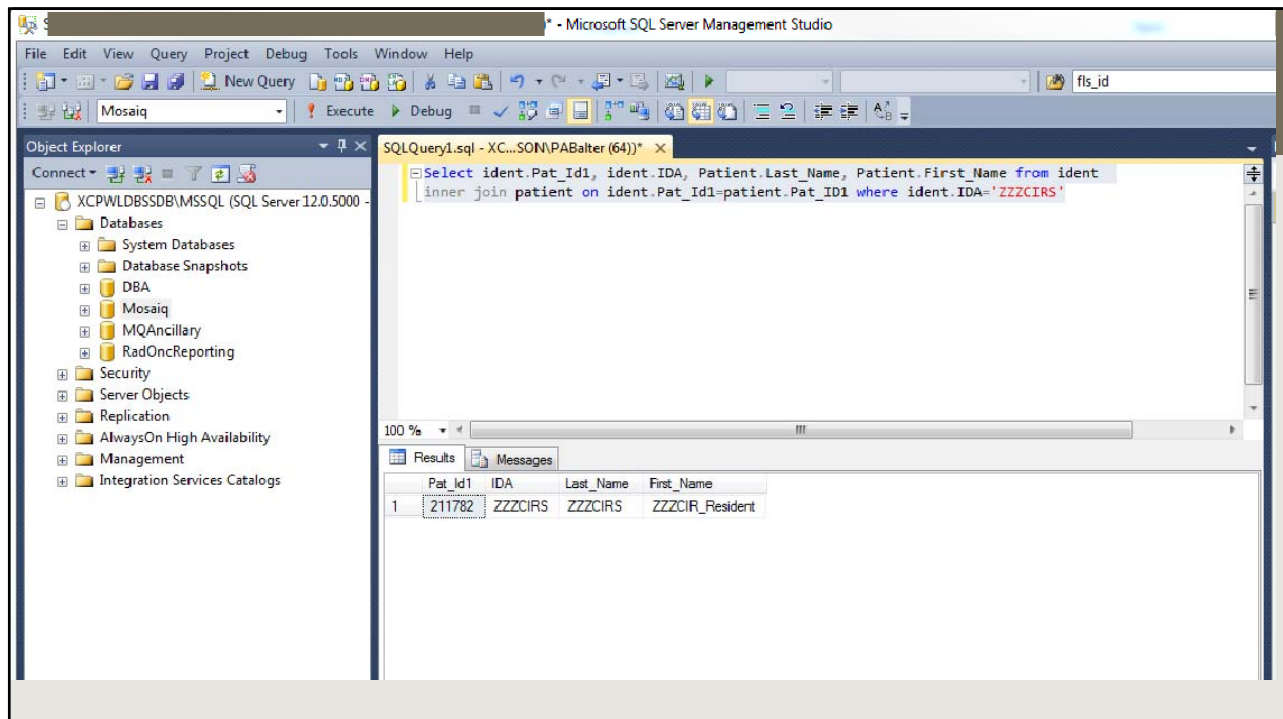
Find patients internal ID based on their Hospital Medical Record Number and get their name as well (stored in another table)

Select ident.Pat_Id1, ident.IDA, Patient.Last_Name, Patient.First_Name
from ident

inner join patient on ident.Pat_Id1=patient.Pat_ID1

where ident.IDA='ZZZCIRS'

Implemented in MicrosoftSQL Management Studio (but should work in any query engine that supports SQL)



The screenshot displays the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the server structure, including the 'Mosaik' database. The central query editor contains the following SQL query:

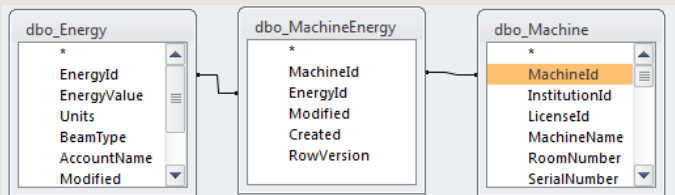
```
Select ident.Pat_Id1, ident.IDA, Patient.Last_Name, Patient.First_Name from ident  
inner join patient on ident.Pat_Id1=patient.Pat_ID1 where ident.IDA='ZZZCIRS'
```

The Results pane at the bottom shows the output of the query, which is a single row of data:

Pat_Id1	IDA	Last_Name	First_Name
211782	ZZZCIRS	ZZZCIRS	ZZZCIR_Resident

Example Join Query (Sun Nuclear Atlas)

- If we want to know what energies each machine has we need to use joins since that information spans 3 tables
- `SELECT dbo_Machine.MachineName, dbo_Energy.EnergyValue, dbo_Energy.Units FROM (dbo_Machine INNER JOIN dbo_MachineEnergy ON dbo_Machine.MachineId = dbo_MachineEnergy.MachineId) INNER JOIN dbo_Energy ON dbo_MachineEnergy.EnergyId = dbo_Energy.EnergyId ORDER BY dbo_Machine.MachineName;`



MachineName	EnergyValue	Units
2100IX	18	MV
2100IX	16	MeV
2100IX	20	MeV
2100IX	12	MeV
2100IX	6	MV
2100IX	9	MeV
2100IX	6	MeV
2103	6	MV
2103	18	MV
2103	20	MeV
2103	6	MeV
2103	16	MeV
2103	12	MeV
2103	9	MeV
2104	9	MeV

Example done using Microsoft Access to Query SQL Database

Queries and SQL using aggregate functions

- Queries can return summary data (SQL aggregation functions)
 - Count, Sum, Max, Min, Avg, etc
 - Example: `Select avg(<field A>) from <table> where <criteria>`
Gives the average value of <field A>
- Generally used with the “GROUP BY” clause to define what set of data is being aggregated
 - Example: `Select <field B>, avg(<field A>) from <table> where <criteria> group by <field B>`
Gives the average value of <field A> for each <field B>

Example aggregate Query (Sun Nuclear Atlas)

- We can use aggregate functions if we want to know how many of each energy we have across all machines:
- SELECT dbo_Energy.EnergyValue,
Count(dbo_Energy.EnergyValue) AS
CountOfEnergyValue FROM (dbo_Machine INNER JOIN
dbo_MachineEnergy ON dbo_Machine.MachineId =
dbo_MachineEnergy.MachineId) INNER JOIN
dbo_Energy ON dbo_MachineEnergy.EnergyId =
dbo_Energy.EnergyId GROUP BY
dbo_Energy.EnergyValue;

EnergyValue	CountOfEnergyValue
2	1
4	3
5	1
6	64
7	3
8	1
9	30
10	7
11	2
12	29
15	9
16	28
18	23
20	26

Example aggregate query using a derived value (Sun Nuclear Atlas)

- Since Energy = 6 can be 6 MV or 6 Mev to further specify we need to build a derived field to sort on that includes both the energy value and the units
- We can query and sort based on this derived value:
EnergyValue] & [Units]

```
SELECT [dbo_Energy]![EnergyValue] & [dbo_Energy]![Units] AS Expr1,  
Count(dbo_Energy.EnergyValue) AS CountOfEnergyValue FROM (dbo_Machine  
INNER JOIN dbo_MachineEnergy ON dbo_Machine.MachineId =  
dbo_MachineEnergy.MachineId) INNER JOIN dbo_Energy ON  
dbo_MachineEnergy.EnergyId = dbo_Energy.EnergyId GROUP BY  
[dbo_Energy]![EnergyValue] & [dbo_Energy]![Units];
```

Expr1	CountOfEnergyValue
10MeV	1
10MV	6
11MeV	2
12MeV	29
15MeV	3
15MV	6
16MeV	28
18MV	23
20MeV	26
2MV	1
4MeV	3
5MeV	1
6MeV	28
6MV	36
7MeV	3
8MeV	1
9MeV	30

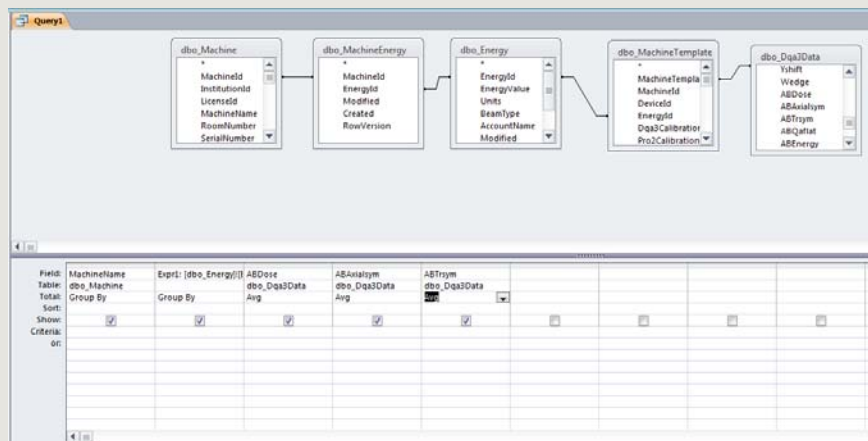
Query Building tools

- Building more useful queries like “what is the average output and symmetry for each machine by energy” can be very complicated
- `SELECT dbo_Machine.MachineName, [dbo_Energy]![EnergyValue] & [dbo_Energy]![Units] AS Expr1, Avg(dbo_Dqa3Data.ABDose) AS AvgOfABDose, Avg(dbo_Dqa3Data.ABAXialsym) AS AvgOfABAXialsym, Avg(dbo_Dqa3Data.ABTrsym) AS AvgOfABTrsym FROM (((dbo_Machine INNER JOIN dbo_MachineEnergy ON dbo_Machine.MachineId = dbo_MachineEnergy.MachineId) INNER JOIN dbo_Energy ON dbo_MachineEnergy.EnergyId = dbo_Energy.EnergyId) INNER JOIN dbo_MachineTemplate ON dbo_Energy.EnergyId = dbo_MachineTemplate.EnergyId) INNER JOIN dbo_Dqa3Data ON dbo_MachineTemplate.MachineTemplateId = dbo_Dqa3Data.MachineTemplateId GROUP BY dbo_Machine.MachineName, [dbo_Energy]![EnergyValue] & [dbo_Energy]![Units];`

MachineName	Expr1	AvgOfABDose	AvgOfABAXialsym	AvgOfABTrsym
2100IX	12MeV	99.7223249300841	-0.533629961683187	0.332782401415182
2100IX	16MeV	99.3277260832826	-0.260144505217171	0.410471441043903
2100IX	18MV	101.851307208846	-3.51383400726538	18.617367738293
2100IX	20MeV	99.578883268125	-0.230178432177204	0.358647870454756
2100IX	6MeV	98.8171913501059	1.1735407609238	0.716443810496579
2100IX	6MV	100.413902185142	4.39687339268117	-20.2291871191576
2100IX	9MeV	99.1991062939368	-6.91842082284848E-02	0.487500558486822
2103	12MeV	99.7223249300841	-0.533629961683187	0.332782401415182
2103	16MeV	99.3277260832826	-0.260144505217171	0.410471441043903
2103	18MV	101.851307208846	-3.51383400726538	18.617367738293
2103	20MeV	99.578883268125	-0.230178432177204	0.358647870454756

Query Building tools

- There are a large number of query building tools that have GUIs that can help you quickly build complicated queries



Queries that change the data

- Some query keywords can be used to change the data in the database
 - Insert: adds rows
 - Update: modifies rows
 - Delete: removes rows
 - Merge: combines rows
- These can be very powerful but:
 - Can unexpectedly modify/remove large amounts of data
 - Should not be used on clinical databases (ARIA, MOSAIQ)

Example Aria: Scripting API for data access

- Many programming languages have built-in support for data access
- The Varian API provides objects\methods\structures to query the ARIA/Eclipse database
 - `// getting all treatment plan's data under "C2" course in selected patient`
 - `var allPlansInfo = from Course c in ThePatient.Courses`
 - `where c.Id == "C2"`
 - `select new`
 - `{ plans = c.PlanSetups,`
 - `Course = c, };`
 - `//query finds the first PTV structure:`
 - `Structure target = (from s in StructureSet.Structures`
 - `where s.DicomType == "PTV"`
 - `select s).FirstOrDefault();`

Amy Liu

Queries that return a large amount of data

- Poorly written (or thought out) queries can adversely affect the performance of the database server the one that is also running your clinic in the case of MOSAIQ or ARIA)
 - Queries should have a “LIMIT” clause during testing if supported by your SQL server
 - Queries should be tested on non-clinical systems first, if possible.
 - If no development system exists consider testing during off-hours
 - For relatively small databases consider making a backup of the database and run queries against the backup during debugging

GUI based Query builders

- There are multiple sets of software both commercial and open source that will provide a framework for helping you build queries.
 - Many will automatically provide joins based on the database scheme
 - They will write SQL that can be manually edited, if needed, and/or copied into code for automated querying
- Examples
 - Microsoft SQL management studio
 - MS Access (used for some of the examples in this work)
 - pgAdmin (PostgreSQL)
 - Many others (check google)

Views, Triggers, and Stored procedures

- Most SQL servers support the several non-date objects in the database used for efficient data access and database integrity:
 - Views: Predefined queries to simplify the users access to the database
 - Stored Procedures: Pre-compiled SQL statements
 - Run faster than normal queries
 - Triggers: Procedures designed to run automatically based on other events
 - Can be used for data integrity checks
 - Often used to create audit logs

Why bother writing you own queries

- Most or all data achievable in the query is available via the vendors GUIs
- Consolidating data from the GUI can be laborious
- Trend analysis, trivial in the database space, can be impractical via GUI
 - Example: How many times in the last year was a film rejected by MD in the last year

```
Select image.Att_App, count(image.Att_App)
from image where (image.Image_Class=9 or
image.Image_Class=13 ) and image.Study_DtTm >
'2017-01-01' group by image.Att_App
```

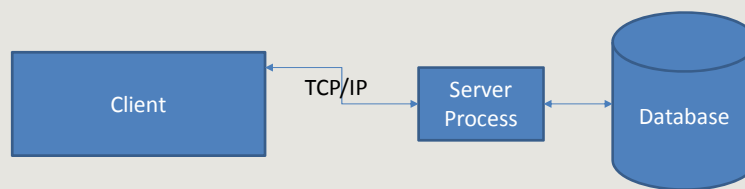
Not Reviewed	4,440	1.2%
Approved	324,541	88.4%
Rejected	38,257	10.4%
Total	367,238	

Reports

- Forms with areas filled in by queries embedded within the form
- Often the only “official” way to get summary data out of many systems
- Are useful when the same data is to be queried and presented multiple times
 - Physics weekly check list from MOSAIQ
 - Billing documents
 - Treatment summaries
- Some database systems have integrated report generators
 - MS Access
 - Microsoft SQL Server Reporting Services (SRSS)
- There are many 3rd party report generators
 - Crystal Reports is a combination query builder and report generated used as the OEM report generator for many systems

The Client Server Database Model

- The database runs as it's own process either on the local computer or on a distant computer
- Applications send transactions to the database
- Transactions maybe processed via an intermediate layer (driver)
 - Microsoft ODBC driver allows many different types of databases to talk to one another



Advantages of the client server model

- Separates database development from application development
- Client applications are independent from physical location of data
- Client systems can be optimized for display and user interface while database server can be optimized for performance
- Reduce traffic on the network as only processed data is transferred
- Can handle concurrent access by multiple users (better than a shared file)

Common Database systems

- Oracle: High performance “main frame” database
- Microsoft SQL: High performance clustered system or local “lite versions” available
- PostgreSQL: Open source database
- MySQL: Another very popular open source database
 - RedCap (front end on MySQL) for “building and managing online surveys and databases”
- MS Access: great general purpose database that hides much of coding.
- SQLite: open source database used as an embedded database in many other applications
- Many many more (just google open source SQL)

Databases in Radiotherapy

- MOSAIQ
 - Aria
 - Sun Nuclear Atlas
 - Raystation
 - FUJI Evercore
 - Radcalc
 - Pinnacle (for a limited amount of the data)
 - Varian RPM
- Microsoft SQL
 - Oracle
 - SQLite
 - PostgreSQL
 - mdb file(MS Access)
-

Note the SQL formalism is general enough that many system can support more than one type of database

Backup

- Database systems have backup utilities that the end user should be able to use
 - Simple systems may be just a file backup
 - Other systems have backup and restore functions within the SQL server workspace
 - Many systems exist for real-time (or near-time) backup between database clusters
 - Many enterprise systems allow “rewind” back to a state at an earlier time
- Work with your vendor to understand what backup systems they support and how to best implement backups in your environment

Audit logs

- Many SQL systems have built in auditing
 - Can tell when the database schema has changed
 - Can tell when data has been
 - Accessed
 - Changed
 - Added
 - Deleted
- Many databases have built in history logs that can be used for auditing
 - MOSIAQ keeps previous versions of many table entries in the database
 - Can produce unexpected query results
 - Some are not easily accessible from the GUI but can be found when directly querying the database – Has helped with some root-cause analysis investigations.



- Great thanks to Michael Kantor for many of the examples in this talk as well as a great expansion of the content
- Thank you to Amy Liu for Eclipse API examples