

제3부 다차원 프로세싱

AGENDA

- 데이터 로딩
- 다차원 연산
- 스타스키마, 집계와 분할
- 다차원 질의

- 운영 시스템과 BI 시스템
 - 운영 시스템(OLTP)
 - BI 시스템
 - 데이터 흐름
 - DW, DW 구축 장점
 - BI 데이터 저장 물리적 장소
 - 사용자가 입력 하는 경우
 - 외부 데이터가 필요한 경우
 - 추출 메타 데이터, BI 메타 데이터 통합
 - 데이터 추출
 - 전용 툴
 - BI 툴
 - 초기 로딩과 주기적 갱신
 - 운영 시스템
 - BI 시스템
- 데이터 희박성
 - 셀이란?
 - 데이터 희박성
- 데이터 유형
- 계층구조와 데이터 입력

데이터 로딩 – 운영 시스템과 BI 시스템

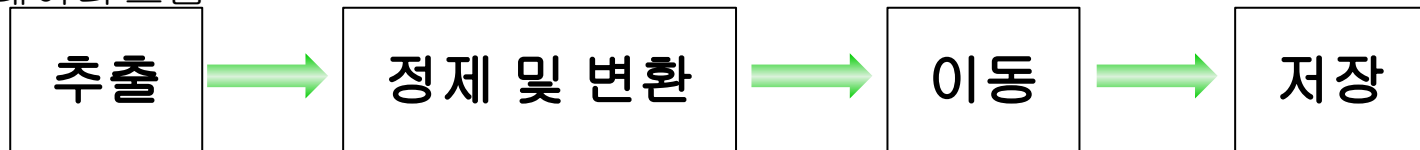
- 운영 시스템(OLTP)

- 기업에서 데이터가 일차적으로 발생하고 수집 되는 곳
- 각 운영 시스템은 특정 비즈니스 업무에 특화되어 있으며 가능한 한 효율적으로 작업을 처리할 수 있도록 조율된다.
- Ex) 은행의 고객과의 거래

- BI 시스템

- 최종 사용자가 다차원 정보에 직접 접근하여 대화식으로 정보를 분석하고 의사결정에 활용하는 곳

- BI 데이터 흐름



- DW, DM 구축 장점
 - 운영 시스템 보호
 - 사용자에게 빠른 응답
 - 통합된 양질의 데이터 제공
 - 쉽게 액세스 가능
- BI 데이터의 물리적 저장 장소
 - 관계형 데이터베이스
 - 다차원 데이터 베이스
- BI 시스템에 사용자가 데이터를 입력하는 경우
 - 예측치, 계획 데이터
 - 예산편성, 사업계획 수립 데이터
 - What-If 분석 등의 다이나믹한 데이터

데이터 로딩 – 운영 시스템과 BI 시스템

- 외부 데이터가 필요한 경우
 - 환율정보, 주가정보 등
- 추출 메타 데이터와 BI 메타데이터의 통합
 - 전용 툴
 - 보다 간편하고 신속하게 데이터를 추출, 가공, 통합, 관리
 - BI 툴
 - 데이터 추출 및 로딩에 필요한 기본적 기능 제공
- 초기 로딩과 주기적 갱신
 - 운영 시스템
 - 수시로 갱신
 - BI 시스템
 - 초기로딩
 - 시스템이 구축되고 필요한 데이터가 최초로 로딩 되는 것
 - 주기적 갱신
 - 모델의 성격에 따라 일별, 주별, 월별 등과 같은 특정 주기에 따라 갱신

데이터 로딩 – 데이터 희박성

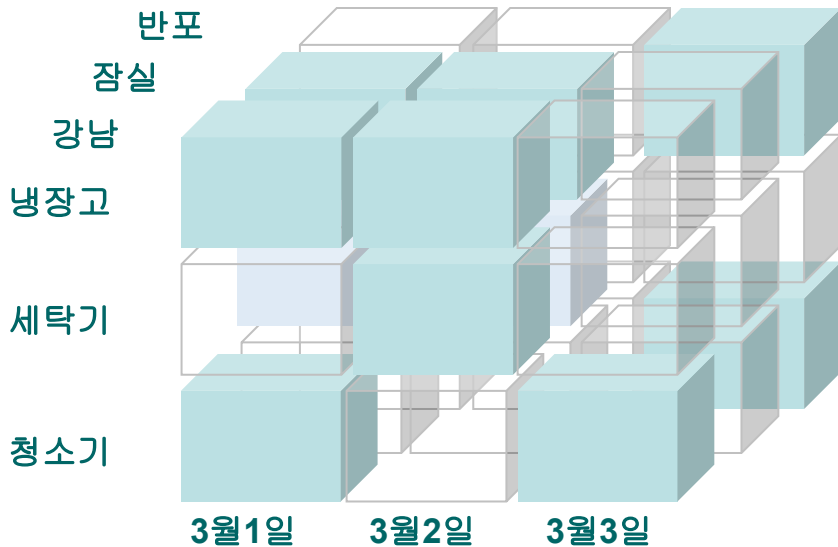
- 데이터 희박성

- 셀이란?

- 데이터가 입력되고 저장될 수 있는 공간

- 데이터의 희박성

- 모든 셀들이 데이터를 가지는 경우는 드물다.
 - 일반적으로 모델을 구성하는 차원수가 많아짐에 따라 희박성은 커진다.



데이터 로딩 – 데이터 희박성

- 희박성이 최소화된 사실 테이블
 - 레코드 수와 인덱스가 많이 필요
 - 사실이 각 행에 표현될 경우 SQL문의 사용은 매우 힘들어 지게된다.

매장	제품	기간	변수	데이터
매장1	제품1	기간1	사실1	450
매장1	제품1	기간1	사실2	100
매장1	제품1	기간2	사실2	120
매장1	제품2	기간2	사실1	530
매장2	제품2	기간2	사실1	510
매장2	제품2	기간2	사실2	80

- 스타스키마와 희박성

매장	제품	기간	사실1	사실2
매장1	제품1	기간1	450	100
매장1	제품1	기간2	-	120
매장1	제품2	기간2	530	-
매장2	제품2	기간2	510	80

데이터 로딩 - 데이터 유형, 계층구조와 데이터 입력

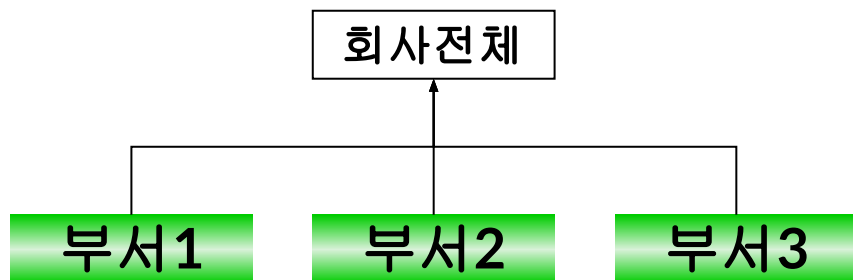
- 데이터 유형

- 일반적으로 셀에 입력되는 데이터의 유형(Type)은 수치 데이터 이다.

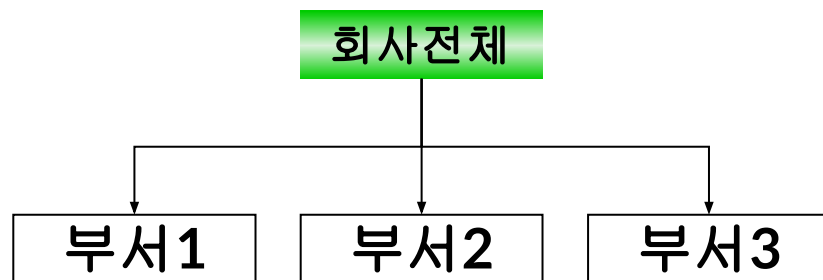
- 계층구조와 데이터 입력


- 데이터가 직접 로딩되는 셀들은 일반적으로 계층구조에서 리프항목(상세항목)들의 조합에 의해 생성되는 셀들이다.
- 경우에 따라 다른 다양한 레벨에 속하는 항목들에 데이터가 입력되기도 한다.

(A) 급료



(B) 임대료



 데이터 입력

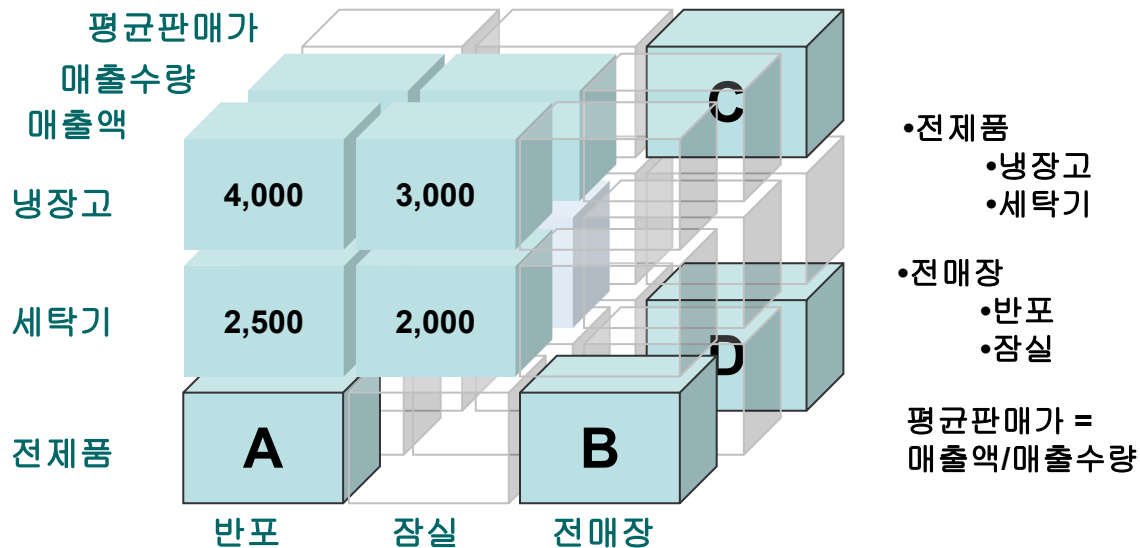
다차원 연산

- 다차원 연산

- 입력된 데이터를 바탕으로 계층구조와 관계식, 애트리뷰트를 참조하여 새로운 데이터를 계산해 내는 과정

- 큐브 내의 셀이 데이터를 갖는 경우

- 입력된 데이터
 - 집계
 - 관계식에 의한 계산



[그림 8-1]

다차원 연산 – 연산순서

● 다차원 연산의 특징

- 각각의 항목들을 중심으로 수행
- 큐브 내의 한 셀은 큐브를 구성하는 각 차원들의 항목이 조합되어 표현
- 각각의 항목들이 각기 다른 룰을 가질 수 있다.
- 각각의 항목들이 가진 룰은 하나의 셀 값을 계산해 내는데 서로 다른 결과를 야기 할 수 있음

매출액		냉장고	세탁기	전제품
	반포	4,000	2,500	6,500
	잠실	3,000	2,000	5,000
	전매장	7,000	4,500	11,500

[그림8-2] 셀 B의 계산

냉장고

	매출액	매출수량	평균판매가		
반포	4,000	200	20		
잠실	3,000	190	20		
전매장	7,000	350	40	또는	20

[그림8-2] 셀 B의 계산

다차원 연산 – 연산순서

- 연산순서의 결정 방식
 - 연산 순서를 정하는 방식
 - 그림 8-4 참조
 - 각각의 룰에 가중치 부여하는 방식
 - Ex) Analysis service의 Solve Order
 - 실제 하나하나의 셀에 대해 어떤 계산식을 적용할지 평가하는 것보다 훨씬 빠르게 계산 할 수 있다.

	매출액	매출수량	평균판매가
반포	4,000	200	
잠실	3,000	190	
전매장	7,000	350	

	매출액	매출수량	평균판매가
반포	4,000	200	20
잠실	3,000	190	20
전매장	7,000	350	20

[그림8-4] 연산순서를 정하는 방식

- 연산시점 종류

- 사용자 질의가 발생하는 시점에 직접 수행
 - 사용자 질의에 신속하게 응답하기 어려움
 - 많은 시스템 자원 소비
- 사용자 질의가 발생하기 전에 미리 수행되어 결과 저장
 - 동일한 사용자 질의에 한번 계산해서 결과를 저장하는 것이 타당함

- 사전연산과 데이터량의 증가

- 사전 연산을 수행하는 방법
 - 필요한 모든 연산을 미리 다 수행하여 결과를 저장
 - 사용자 질의에 최상의 응답성능 제공
 - 로딩된 데이터 양에 비해 연산된 데이터가 많이 커짐(데이터 폭발)
 - 일부만 연산하여 결과를 저장
- 차원이 증가함에 따라 데이터는 지수적 증가 양상을 보인다.
- 팽창계수와 차원수, 희박성의 관계
 - 레벨수↑, 차원수↑, 희박성↑=> 팽창계수↑↑↑
 - 일반적으로 5차원 이상의 큐브에 대해 팽창계수는 2.0을 넘게되고, 차원이 추가될 때마다 0.1씩 증가

- 데이터 폭발을 막는 법
 - 일차적으로 큐브의 설계 시 큐브를 구성하는 차원수, 희박성을 최소한으로 유지
 - 큐브가 설계된 후 큐브의 모든 셀을 미리 계산하지 않고, 되도록 많은 부분을 사용자 질의가 발생하는 시점에서 실시간으로 계산
- 사전 연산 영역의 결정
 - 사전 계산 대상
 - 질의 시 계산하기에 많은 시간이 소요되는 부분
 - 사용자가 자주 요청하는 부분
 - 다른 연산에서 자주 참조되는 부분
 - 전체 집계가 필요한 경우
 - 짧은 시간 내에 연산이 수행 될 수 있는 적은 규모
 - 질의응답성능이 아주 중요한 경우
 - 동시 사용자 수가 많은 경우
 - 최적의 사전연산 영역 관리법
 - 사용자 질의를 로그로 관리하고 이를 바탕으로 많이 쓰이고 오래 걸리는 연산을 파악하여 지속적으로 유지 관리

스타스키마, 집계와 분할 - 집계

- 스타스키마의 특징

- 하나의 모델에는 하나의 사실 테이블이 존재
- 각 차원은 하나의 테이블로 표현, 하나의 칼럼을 기본 키로 가짐(의미 없는 정수값)
- 각 차원 테이블의 기본 키는 사실테이블에 대응되는 칼럼 가짐

- 집계(Aggregation)와 분할(Partitioning)

- 스타스키마 설계에서 응답성능을 향상시키고 유지보수를 용이하게 하기 위해 가장 빈번하게 이용되는 기법

- 집계(Aggregation)

- 스타스키마에서 사전연산은 대부분 집계 작업을 의미함
- 스타스키마에서 집계는 상세 수준의 데이터를 특정 애트리뷰트에 따라 미리 요약하는 작업 (집계작업)
- 일상적으로 액세스 되는 데이터에 대해서는 사전에 요약작업을 하는 것이 일반적
- 많은 BI 시스템이 미리 계산된 요약데이터를 적절히 이용함으로써 수십 배에서 수천 배에 달하는 성능향상을 얻고 있음
- 계층 구조의 상위에 있을 수록 요약 정도가 높아 데이터 양이 적어진다

- 집계 영역의 결정

- 일반적으로 사용자가 자주 필요로 하는 부분을 집계하는 것이 효과적
- 사용자의 요청패턴을 파악하기 위해 집계 데이터 생성 후에도 모니터링 작업 필요
- 많은 레코드가 요약 될 수 있는 애트리뷰트를 중심으로 집계작업 수행 시 집계 데이터로부터 큰 효과 얻음

스타스키마, 집계와 분할 - 집계

- 집계데이터의 저장 방식
 - 별도의 사실 테이블 사용
 - 집계 데이터를 저장하기 위해 새로운 사실 테이블을 생성하는 방법
 - 동일한 사실 테이블에 함께 저장
 - 집계가 이루어지는 차원 테이블에 레벨(Level)필드를 추가하고 원래의 사실 테이블에 기본 데이터와 집계 데이터를 함께 저장하는 방식
- 스노우 플레이크 스키마와 집계
 - 정규화된 스노우 플레이크 스키마
 - 차원 테이블이 정규화되어 있기 때문에 집계 테이블만 생성하면 됨
- 집계 데이터 만드는 이유
 - 질의에 대한 응답속도를 향상시키기 위함
- 집계 데이터 사용시 유의점
 - 주기적으로 집계데이터를 만들고 모니터링
 - 자주 사용하지 않는 집계 데이터는 폐기
 - 많은 CPU자원과 디스크 공간 요구
 - 지속적으로 모니터링 하지 않으면 잘못된 집계 데이터 유지 될 수 있음

스타스키마, 집계와 분할 - 분할

- **분할**

- 스타스키마에서 사실테이블이 매우 큰 경우 사실 테이블을 분리 할 수 있음
- 분할된 테이블들을 애플리케이션은 마치 하나의 테이블처럼 사용

- **분할의 이점**

- 데이터를 논리적인 단위로 나눠 질의에 응답하기 위해 검색해야 하는 레코드 개수를 줄이고 응답성능 향상
- 데이터의 백업 및 복구, 폐기작업을 용이하게 하며 병렬처리의 효과를 잘 이용 할 수 있음
- 데이터 로딩과 같은 배치 작업에 유연성 제공
 - 관련된 테이블만 영향을 받고 나머지 테이블은 정상적으로 사용됨

- **분할의 특징**

- 이점이 큰 대신 추가적인 유지보수 노력이 역시 상대적으로 증가
- 사용자 질의가 분할된 다수의 테이블로부터 데이터를 요청 할 경우 조인, 유니온 필요
- 분할된 테이블을 기술하기 위해 메타데이터가 요구됨
- 어떤 테이블이 사용자가 원하는 데이터를 갖고 있는 지 파악하기 위해 더욱 지능적인 질의문 생성 필요
- 정형화 되지 않은 다양한 질의가 수행되는 역동적인 환경에서 지속적인 조율 필요

스타스키마, 집계와 분할 - 분할

- 분할 시점

- 사실 테이블이 매우 클 때
- 대부분의 질의를 하나의 분할된 테이블만으로 처리할 수 있을 때

- 분할의 종류

- RDBMS에서 지원하는 DB 수준의 분할
- 애플리케이션 수준의 분할
 - 수평 분할
 - 수직 분할

- 수평 분할

- 수평분할은 하나 이상의 비즈니스 차원에 따라 행 수준에서 사실테이블을 분할
- 수평 분할의 특징
 - 가장 일반적인 데이터 분할 방식
 - 많은 관계형 DBMS에 의해 지원
 - 최대의 효과를 얻기 위해서는 안정적인 애트리뷰트를 중심으로 분할 해야 함
 - 애트리뷰트: 기간 차원의 월, 분기 등은 안정적인 애트리뷰트
 - [그림 9-9] 참조

스타스키마, 집계와 분할 - 분할

기간키	매장키	제품키	매출액
990101	110	1101	27800
990103	120	1200	39000
990202	210	1200	27800
990208	120	1201	39000
990304	210	1301	30300
990307	220	1101	21300

기간키	매장키	제품키	매출액
990101	110	1101	27800
990103	120	1200	39000

기간키	매장키	제품키	매출액
990101	110	1101	27800
990103	120	1200	39000

기간키	매장키	제품키	매출액
990101	110	1101	27800
990103	120	1200	39000

기간키	매장키	제품키	매출액
990101	110	1101	27800
990103	120	1200	39000

[그림 9-9] 수평분할

– 수직분할

- 수직분할은 열을 기준으로 사실테이블을 쪼개는 것 (그림 9-10 참조)
- 특징
 - 사실 테이블의 기본 키가 중복되기 때문에 전체적인 저장공간은 더 많이 차지
 - 수직분할은 사용자 집단 별로 관심 있는 칼럼들이 다르거나 자주 액세스 되지 않는 칼럼들이 있을 경우 이들 칼럼을 별도의 테이블에 저장함으로써 테이블의 크기 줄임
 - 따라서 검색해야 하는 하드 디스크 양이 줄어듦
- 필요이유
 - ‘사실’들간에 희박성이 크게 차이 나는 경우
 - 갱신 주기가 다른 경우
 - ‘사실’들이 개수가 많은 경우 물리적 칼럼 제한 피함

[그림 9-10] 수직분할

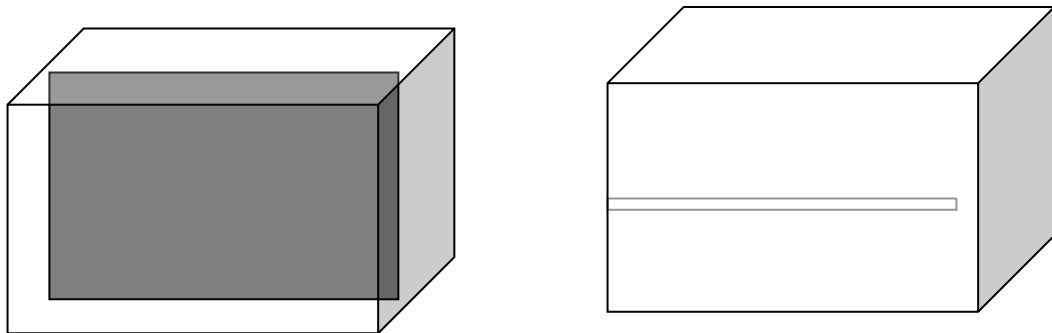
기간키	매장키	제품키	매출액	매출원가	운송비
990101	110	1101	27800	12100	780
990103	120	1200	39000	15500	810
990202	210	1200	27800	13000	810
990208	120	1201	39000	14500	930
990304	210	1301	30300	14300	840
990307	220	1101	21300	14530	900

기간키	매장키	제품키	매출액
990101	110	1101	27800
990103	120	1200	39000
990202	210	1200	27800
990208	120	1201	39000
990304	210	1301	30300
990307	220	1101	21300

기간키	매장키	제품키	매출원가	운송비
990101	110	1101	12100	780
990103	120	1200	15500	810
990202	210	1200	13000	810
990208	120	1201	14500	930
990304	210	1301	14300	840
990307	220	1101	14530	900

다차원 질의 – 슬라이싱(Slicing)과 다이싱(Dicing)

- 사용자는 주요 비즈니스 항목들을 다양한 각도에서 조회 하고 자유롭게 비교할 수 있다. 이러한 조회작업을 다차원 질의(**Multi-Dimensional Query**)라 한다.
- 다차원 질의의 기본은 사용자가 큐브의 어떤 부분을 볼 것인지 정의하는 것이다.
- 다차원 질의는 [그림 10-1]에서 보는 것처럼 마치 사용자가 큐브의 일부분을 자신이 원하는 형태로 절단하여 살펴보는 것에 비유 할 수 있다.
- 이러한 이유로 다차원 질의를 슬라이싱과 다이싱 이라고 부른다.



[그림 10-1] 슬라이싱과 다이싱

다차원 질의 – 슬라이싱과 다이싱

- 큐브와 보고서 차원
 - 많은 BI툴 중에서 큐브를 구성하는 하나의 차원은 보고서의 행, 열, 페이지 중 오직 하나만을 구성할 수 있다.
 - 다차원 모델을 설계하면서 어떤 형태의 보고서가 필요한지 미리 고려할 필요가 있다.

[그림 10-3] 모델차원과 보고서 차원

반포, 1월, 매출액

	블랙	화이트
냉장고	520	110
청소기	470	160
청소기	320	70

제품차원

제품차원

다차원 질의 – 슬라이싱과 다이싱

- 차원의 중첩(Nesting)

- 보고서의 행이나 열이 두 개 이상의 차원으로 구성될 경우 Nesting이 일어난다고 한다.

[그림 10-5] 차원의 중첩

(A)

		1월	2월	3월
냉장고	매출액	320	250	300
	매출원가	70	85	80
세탁기	매출액	470	500	450
	매출원가	160	150	140
청소기	매출액	520	410	430
	매출원가	110	130	120

(B)

	1월		2월		3월	
	매출액	매출원가	매출액	매출원가	매출액	매출원가
냉장고	320	70	250	85	300	80
세탁기	470	160	500	150	450	140
청소기	520	110	410	130	430	120

다차원 질의 – 슬라이싱과 다이싱

- **피보팅(Pivoting)**

- 사용자는 보고서의 행, 열, 페이지 차원을 무작위로 바꾸어 볼 수 있다.
- 정형화 되지 않은 자신이 원하는 형태의 다양한 보고서를 역동적으로 만들어 볼 수 있다.

- **항목의 선택/ 필터링**

- **배경**

- 사용자가 특별히 조회 하고픈 몇 개의 항목들을 조회 하기 위한 과정에 시간이 걸린다.

- **정의**

- 보고서 상에 나타나는 데이터를 특정 기준에 부합하는 항목으로 한정하는 것
- 사용자는 계층구조나 애트리뷰트 등을 활용하여 보다 효과적으로 원하는 항목을 선택 할 수 있다.

- **선택/ 필터링 4가지 기준**

- 계층구조에 의한 선택
- 애트리뷰트에 의한 선택
- 항목 이름에 의한 선택
- 데이터에 기초한 선택
 - 많은 시스템 자원을 필요로 한다.

다차원 질의 – 슬라이싱과 다이싱

- 값을 갖지 않는 셀의 추출
 - 값을 갖지 않는 셀은 '0' 값을 갖는 셀과 구분 되어야 한다.
- 계층구조와 항목간 관계의 사용자 설정
 - 사용자가 직접 작성하는 룰에 의해서도 수행될 수 있다.

다차원 질의 – 드릴다운, 드릴업, 드릴어크로스, 드릴쓰루

방대한 분량의 데이터에 접근하는 기본적인 방식으로 드릴다운, 드릴업, 드릴어크로스, 드릴쓰루가 있다.

- 드릴다운(Drill Down)

- 요약된 형태의 데이터 수준에서 보다 구체적인 내용의 상세 데이터로 단계적으로 접근
- 드릴다운과 드릴업의 경로는 모델의 계층구조를 따른다.
- 반드시 계층구조 내의 경로에 따라서만 일어나는 것은 아니다.

- 드릴어크로스

- 다른 큐브의 데이터에 접근하는 것을 말한다.

- 드릴쓰루

- BI시스템으로부터 데이터 웨어하우스 혹은 OLTP시스템에 존재하는 상세 데이터에 접근하는 것을 말한다.
- BI 시스템에 존재하지 않는 데이터가 추가적으로 필요할 경우 자동적으로 데이터 웨어하우스 나 OLTP상의 데이터에 접근할 수 있는 통로를 제공한다.
- 리치쓰루(Reach Through)라 불리기도 한다.

다차원 질의 – 소팅과 랭킹

데이터 분석 과정에서 사용되는 가장 일반적인 기법

- 차원이 중첩된 경우
 - 다차원 질의에서 차원의 중첩이 이루어진 경우 이처럼 차원의 중첩을 인식하여 소팅이 수행되면 훨씬 효과적인 분석이 이루어 질 수 있다.
- 다양한 레벨이 존재하는 경우
 - 항목들 간의 계층구조를 인식하여 소팅이 이루어질 수도 있다.

[그림 10-11] 계층구조를 인식한 소팅

(A)

	매출액	매출수량
수도권	9851	714
강북권	6550	481
명동	3570	250
강남권	3301	233
종로	2980	231
서초	1213	87
잠실	1106	79
반포	982	67

(B)

	매출액	매출수량
수도권	9851	714
강북권	6550	481
명동	3570	250
종로	2980	231
강남권	3301	233
서초	1213	87
잠실	1106	79
반포	982	67

다차원 질의 - BI 조인

- 다차원 질의는 하나 이상의 큐브로부터 데이터를 요청할 수 있다.
- 질의 과정에서 다수의 큐브를 논리적으로 조인할 필요가 있다. 이것이 **BI 조인**이다.

쿼리 툴과 리포팅 툴에서도 관계형 데이터소스에 덧붙여 다차원 데이터를 조회해 볼 수도 있도록 기능을 향상 시키고 있다.

- **스프레드시트와 BI**
 - 다차원 질의를 수행하기 위해 다차원 질의 전용 툴 이외에도 스프레드시트가 많이 사용된다.
- **웹과 BI**
 - 현재 정보를 전달하는 가장 효과적인 수단으로 각광 받고 있다.

BI 시스템은 많은 정보를 사용자가 보다 쉽고 활용할 수 있고 다양한 각도에서 역동적으로 분석할 수 있도록 제공한다.

- 예외사항 보고기능

- 예외적인 항목을 쉽게 파악할 수 있도록 지원한다.
- 예외사항을 알려주기 위해 주로 시각적인 효과가 이용된다.

- 에이전트

- 사용자를 대신하여 백그라운드에서 작업을 수행하는 독립적인 소프트웨어 객체
- BI 시스템에서 관심을 끌게 된 이유
 - 사용자들이 자주 수행하는 질의와 조건들 중 많은 부분이 반복적이다.
 - 정보과잉 문제를 해결하기 위해 필수적이다.
 - 사전에 정의된 특정 사건에 의해 자동적으로 구동 될 수 있다.
 - 에이전트는 BI데이터베이스에 무엇이 발생했는지 지속적으로 모니터링하고 예외사항이 발생한 경우 이를 사용자에게 알려준다.
- 장점
 - 데이터분석과 필터링 과정의 일부를 에이전트에 위임함으로써 BI시스템은 더 완벽한 정보 전달 메커니즘이 될 수 있다.