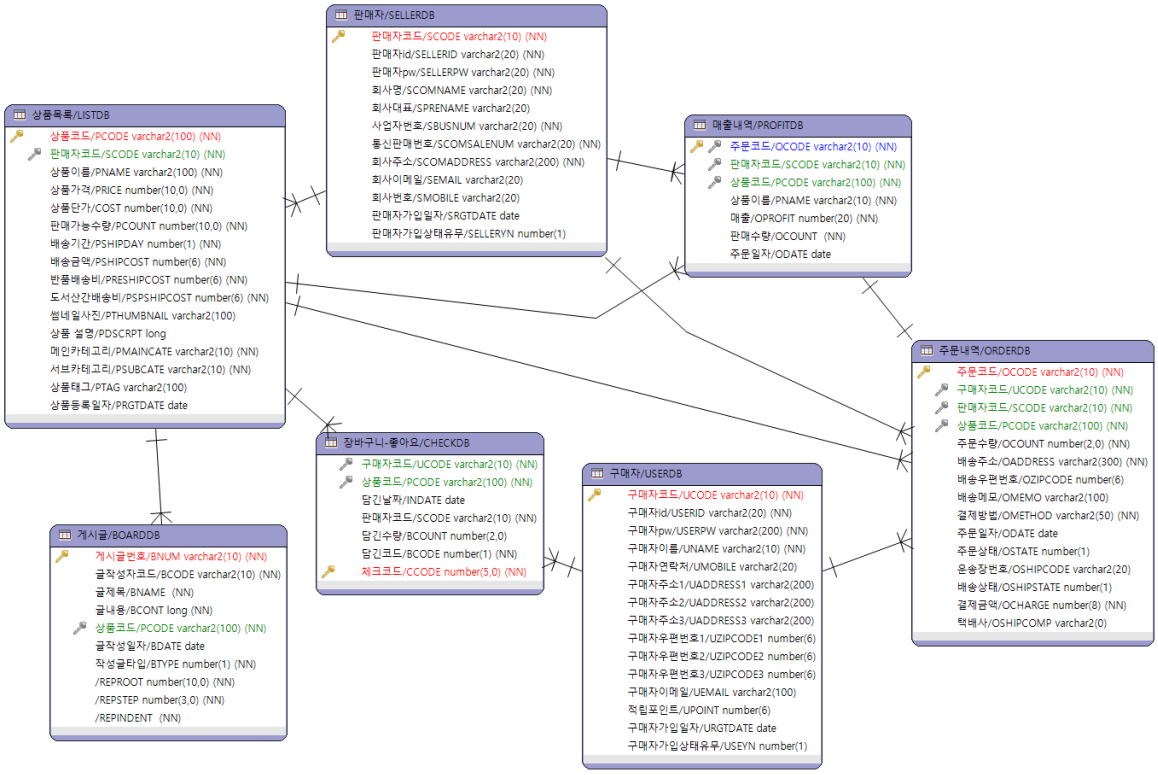
TeamProject_OnlineStore : DB

0. 개요

- 팀프로젝트에서 조장을 맡아 작업을 진행하며 DB와 쿼리문 작성을 모두 맡아 했습니다.
- 그 과정과 이유를 나름대로 정리해보았습니다. 제출이 아닌 개인적으로 복습을 하고 공부를 하기위해 작성했던 문서입니다.

1. 요구사항 정리 및 DB 구조 짜기

- 팀프로젝트로 진행할 주제를 정한 후, 레퍼런스 할 사이트를 정해 해당 사이트의 구조를 분석하며, Table을 구분하지 않고 칼럼이 필요한 항목들을 정리함.
- 각자 구현할 파트를 분배하고 이를 기반으로 기능별로 파트를 나눠 Table을 할당한 후, key가 될 수 있는 칼럼을 선별한 후 불필요하게 중복된 칼럼이 생기지 않도록 고려하여 Table과 칼럼을 정리함.
- 조원들이 메모와 변동사항을 수시로 확인할 수 있도록 엑셀로 파일을 만들어 공유함. (https://1drv.ms/x/s!AoAFrj9GbibY93GHNb8wjXn2KXWx?e=Otku9q)



(페이지 제일 하단에 큰 이미지로 재삽입되어있습니다.)

2. 작업을 하며 아쉬웠던 점

DB의 기본적인 구조(Table과 필요한 칼럼 등)를 가이드라인으로 삼고자 기본적인 DB 작업을 해서 조원들에게 제안함. 모두 확인과 검토를 하였고 각자의 요구사항에 맞추어 수정을 했으나, 모든 조원들이 각자의 파트를 어떻게 구성하여 어떻게 구현할지를 모두 다 정하고 작업을 시작하지 못하여 코드를 작성한 후 작은 수정들이 지속적으로 발생하게 됨.

- Pname(상품이름)이 ListDB(상품DB)에 존재함에도 불구하고 ProfitDB(매출DB)에 이중으로 존재하게 된 점.

  ◌ ProfitDB에 pcode(상품코드) 칼럼이 있으며, ListDB는 pname(상품이름) 칼럼을 가진 pcode를 PK로 쓰는 테이블이었음에도 구조를 짜는 과정에서 미처 발견하지 못한 것이 아쉽다.

  ◌ 그 과정에서 ProfitDB가 단독으로 존재할 필요가 있는지에 대해 고민함. ProfitDB는 단독으로 데이터를 갖는 것이 아니라, OrderDB에서 ostate(주문상태)가 5(구매확정 번호)인 데이터만을 가져와서 판매자에게 제공하는 매출내역 DB라서, 언제든 다른 DB를 join해서 데이터를 조회해올 수 있다고 생각해 의문을 가지게됨. 하지만 ListDB(상품DB)에서 cost(단가)와 price(판매가)의 차이를 계산하기 때문에 해당 DB에 update가 이루어지면, 이전 판매 내역까지 oprofit(판매가-단가±택배비로 계산되는 매출칼럼) 역시 판매 당시의 값이 아닌, 변동된 값으로 변하기 때문에 별도의 DB로 존재하는 것이 맞다고 결론내림.

  ◌ 해당 칼럼을 최종적으로 삭제하지 않게 된 이유는,

  ```
  insert into profitdb
  (select o.ocode, l.scode, l.pcode, l.pname,
   case when o.ozipcode = (도서산간지역)
          then ((l.price*o.ocount)+l.pshipcost+l.pspshipcost)
          else ((l.price*o.ocount)+l.pshipcost) end,
  o.ocount, o.odate
  from orderdb o left outer join listdb l on (o.pcode = l.pcode));
  ```

  라는 쿼리로 작동하는, 다른 DB에 이미 추가되어 더 이상의 update가 없는 데이터만을 기반으로 insert되고 단독으로 update, delete되는 일이 없는 테이블이고, 다른 조건 없이 테이블의 데이터를 모두 조회하는 페이지가 있었으며, 해당 페이지의 조회량이 많아질 경우 join하는 것 보다 단독조회의 효율이 더 좋을 수 있다고 생각하여 수정을 하지 않게 됨. 하지만 처음에는 oprofit에 들어가는 case-when 문장이 (((l.price-l.cost)*o.ocount)+(상황에 따른 택배비)) 였으나, 해당 부분의 구성이 변하게되며 상단의 쿼리로 바뀌게 됨.

- 초반 요구사항을 정리하는 과정에서 페이지를 어떻게 구성할지 다 정하지 못한 조원이 있었고 또 구상을 한 후에도 작업 과정에서 변동이 생겨 칼럼의 크기를 정확하게 하지 못해 수정이 있었던 점

  ○ OrderDB(주문DB)에 omethod(결제방법) 칼럼에 '실시간 계좌이체', '무통장 송금' 과 같은 방식으로 입력하게 되어 칼럼의 크기를 늘림.

  ○ UserDB(구매자DB)의 userpw(비밀번호) 칼럼에 단순 pw를 받는 것이 아닌 암호화를 해 넣게 되면서, 칼럼의 크기를 늘림.

  ○ 구조를 짜는 동안 CheckDB에 PK칼럼이 필요하다는 것을 인식하지 못해, PK가 되어줄 칼럼 없이 작업을 함. 동일한 제품을 장바구니에 중복하여 담았을 경우 FK인 pcode(상품코드), ucode(구매자코드)로 update와 delete를 하면 모든 항목이 수정, 삭제되는 상황을 마주함.

  이미 동일한 pcode를 가진 상품이 CheckDB에 존재할 경우 insert가 아닌 update가 되게 하거나 PK를 추가해야 했고, 모든 table에는 PK가 있어야 한다는 교과서의 말을 따라 SEQ를 사용하는 PK ccode를 추가함.

  ○ UserDB의 upoint(적립금)칼럼을 최종적으로 구현하지 않아, 사용하지 않는 칼럼이 됨.

  초반 구상을 할 때 추가 할 수 있을 거라고 생각한 많이 어렵지 않은 기능이 시간에 쫓겨 더 필요한 부분들에게 우선순위가 밀려 구현하지 못한 것이 아쉽지만, DB에서 기능구현을 못해 사용하지 못 한 칼럼은 upoint 하나 뿐이고, 회원가입시 default 0 으로 해놓았기 때문에 다른 기능 구현에 문제를 발생시키지도 않았으며, 만약 사이트를 실제로 만들었다면 나중에 필수적으로 추가해야 할 기능이라고 생각해 삭제하지 않고, 나중에 혼자라도 포인트제도를 추가해보기로 함.

- 쿼리문이 더 간단할 수 있지 않았을까 하는 아쉬움.

  ○ 프로젝트에 할당된 기간이 코스 중간에 있어서 스프링 프로젝트 로직을 다 익히지 못한 상태였음. 특히 DAO를 사용하는 법을 익히지 못한 상태였고, 수업은 mvc구조로 진행하면서 Service는 사용하는 구조였음. DAO와 Impl이 없는 jsp – Controller.java – Service.java – Mapper.xml + DTO 구조로 작업을 하였고, 프로젝트를 시작할 때 시간이 많지 않기에 기존에 배운 것을 제대로 익혔는지를 목표로 진행했기에 해당 구조를 그대로 사용함.

  나중에 mvc패턴과 스프링 프로젝트 로직에 조금 더 익숙해지면서, 프로젝트를 진행할 때의 프로젝트 구조가 쿼리문을 더 복잡하게 만든 것 같아서 아쉬움.

  ○ 수업 중에는 아주 간단한 CRUD만을 했기 때문에, 쿼리문이 조금이라도 복잡하고 길어지면 이 문장이 DB에 부담을 주지는 않을까? 하는 고민을 하게 됨.

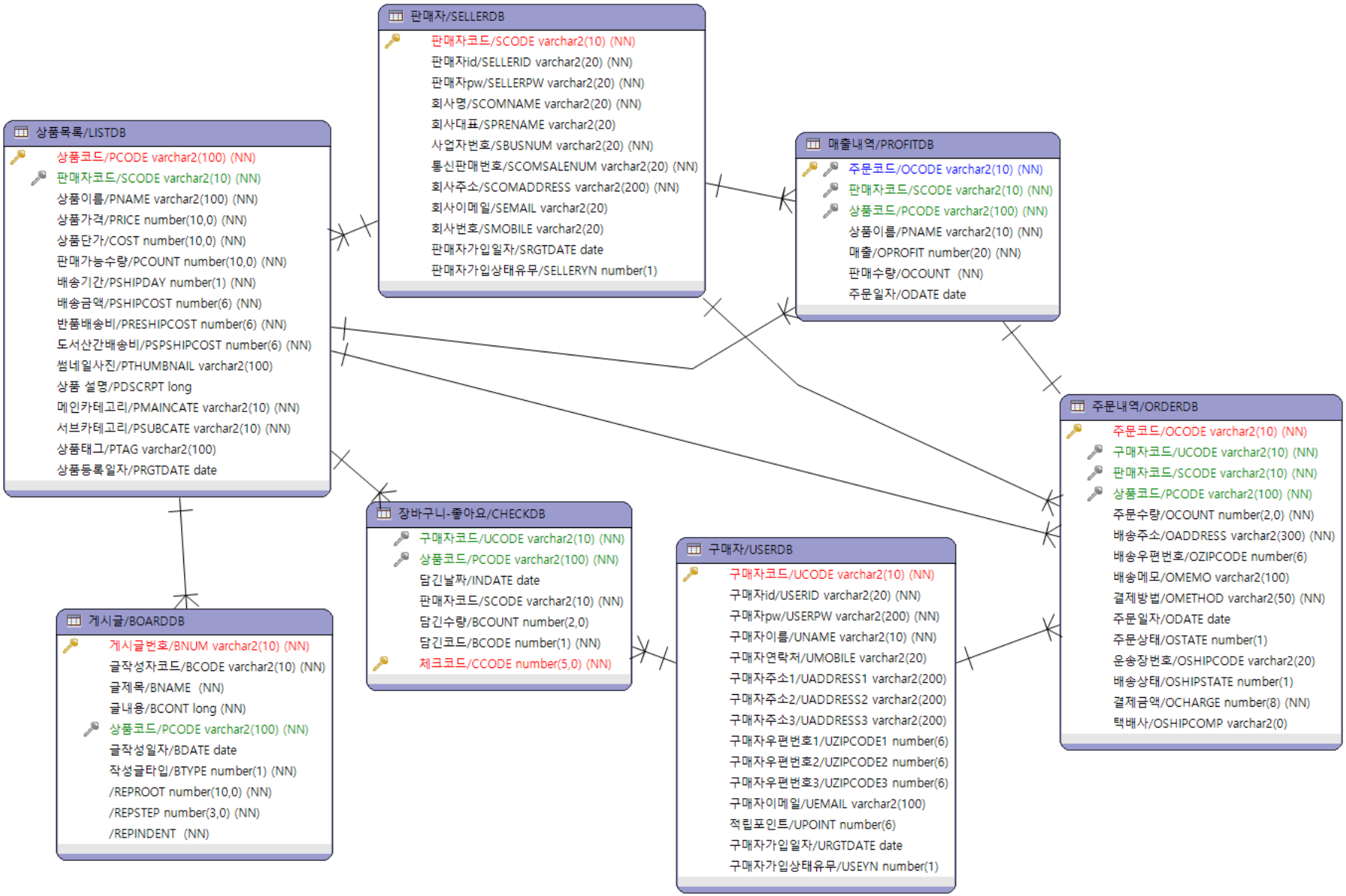
3. 작업하며 생각한 것

- 그냥 쿼리문 짜는게 재미있어서 오라클 책을 틈날때마다 혼자 공부했는데, 책 한권을 딱 마무리한 시점에 팀프로젝트를 시작해서 책에서 공부한 걸 사용해 볼 수 있어서 좋았음.

- 한 조원이 조회의 간편함을 이유료 각 테이블마다 필요한 칼럼을 모두 추가해달라는 요구를 했으나, 그렇게 할 경우 수정/삭제가 이루어질 때 모든 테이블에 동일한 상태를 유지하는 과정에 결점이 생기기 쉽고, 그런 결점이 모든 데이터를 불확실하게 만들 수 있다는 이유로 반려함.
  해당 조원이 쿼리문을 작성하는데 어려움을 겪었기 때문에, 쿼리문을 대신 짜주겠다고했고, 덕분에 내 파트에서는 사용할 일이 없을만한 쿼리 문법들을 사용해 볼 수 있었음.

  하지만 로컬서버에서만 가동해봤기때문에, 이 사이트를 실제 서버로 가동시켰을 때, DB의 CPU 사용과 DB 동시사용을 확인 할 수 없어서 아쉬웠음.

  프로젝트에 판매자가 재고를 등록하면, 재고가 남아있을 때만 구매가 가능하도록 하는 기능을 넣던 중, 재고가 적게 남았을 때 주문이 동시에 들어온다면 쿼리문이 먼저 실행 된 구매자의 주문이 우선시되는지, 선구매처리된 사람의 주문이 DB에 저장된 후, 재고량의 변동이 다음 구매자에게 얼마나 빠르게 반영될 수 있는지 더 알아보고싶음.


판매자/SELLERDB
판매자코드/SCODE varchar2(10) (NN)
판매자id/SELLERID varchar2(20) (NN)
판매자pw/SELLERPW varchar2(20) (NN)
회사명/SCOMNAME varchar2(20) (NN)
회사대표/SPRENAME varchar2(20)
사업자번호/SBUSNUM varchar2(20) (NN)
통신판매번호/SCOMSALENUM varchar2(20) (NN)
회사주소/SCOMADDRESS varchar2(200) (NN)
회사이메일/SEMAIL varchar2(20)
회사번호/SMOBILE varchar2(20)
판매자가입일자/SRGTDATE date
판매자가입상태유무/SELLERYN number(1)
상품목록/LISTDB
상품코드/PCODE varchar2(100) (NN)
판매자코드/SCODE varchar2(10) (NN)
상품이름/PNAME varchar2(100) (NN)
상품가격/PRICE number(10,0) (NN)
상품단가/COST number(10,0) (NN)
판매가능수량/PCOUNT number(10,0) (NN)
배송기간/PSHIPDAY number(1) (NN)
배송금액/PSHIPCOST number(6) (NN)
반품배송비/PRESHIPCOST number(6) (NN)
도서산간배송비/PSPSHIPCOST number(6) (NN)
썸네일사진/PTHUMBNAIL varchar2(100)
상품 설명/PDSCRPT long
메인카테고리/PMAINCATE varchar2(10) (NN)
서브카테고리/PSUBCATE varchar2(10) (NN)
상품태그/PTAG varchar2(100)
상품등록일자/PRGTDATE date
매출내역/PROFITDB
주문코드/OCODE varchar2(10) (NN)
판매자코드/SCODE varchar2(10) (NN)
상품코드/PCODE varchar2(100) (NN)
상품이름/PNAME varchar2(10) (NN)
매출/OPROFIT number(20) (NN)
판매수량/OCOUNT (NN)
주문일자/ODATE date
주문내역/ORDERDB
주문코드/OCODE varchar2(10) (NN)
구매자코드/UCODE varchar2(10) (NN)
판매자코드/SCODE varchar2(10) (NN)
상품코드/PCODE varchar2(100) (NN)
주문수량/OCOUNT number(2,0) (NN)
배송주소/OADDRESS varchar2(300) (NN)
배송우편번호/OZIPCODE number(6)
배송메모/OMEMO varchar2(100)
결제방법/OMETHOD varchar2(50) (NN)
주문일자/ODATE date
주문상태/OSTATE number(1)
운송장번호/OSHIPCODE varchar2(20)
배송상태/OSHIPSTATE number(1)
결제금액/OCHARGE number(8) (NN)
택배사/OSHIPCOMP varchar2(0)
장바구니-좋아요/CHECKDB
구매자코드/UCODE varchar2(10) (NN)
상품코드/PCODE varchar2(100) (NN)
담긴날짜/INDATE date
판매자코드/SCODE varchar2(10) (NN)
담긴수량/BCOUNT number(2,0)
담긴코드/BCODE number(1) (NN)
체크코드/CCODE number(5,0) (NN)
구매자/USERDB
구매자코드/UCODE varchar2(10) (NN)
구매자id/USERID varchar2(20) (NN)
구매자pw/USERPW varchar2(200) (NN)
구매자이름/UNAME varchar2(10) (NN)
구매자연락처/UMOBILE varchar2(20)
구매자주소1/UADDRESS1 varchar2(200)
구매자주소2/UADDRESS2 varchar2(200)
구매자주소3/UADDRESS3 varchar2(200)
구매자우편번호1/UZIPCODE1 number(6)
구매자우편번호2/UZIPCODE2 number(6)
구매자우편번호3/UZIPCODE3 number(6)
구매자이메일/UEMAIL varchar2(100)
적립포인트/UPOINT number(6)
구매자가입일자/URGTDATE date
구매자가입상태유무/USEYN number(1)
게시글/BOARDDB
게시글번호/BNUM varchar2(10) (NN)
글작성자코드/BCODE varchar2(10) (NN)
글제목/BNAME (NN)
글내용/BCONT long (NN)
상품코드/PCODE varchar2(100) (NN)
글작성일자/BDATE date
작성글타입/BTYPE number(1) (NN)
/REPROOT number(10,0) (NN)
/REPSTEP number(3,0) (NN)
/REPINDENT (NN)