**Mini Group Project (Facebook Program)**

**3조(박준철, 김예원, 이호준, 전승윤, 조용진, 연혜원)**

|  |  |  |
| --- | --- | --- |
| 요구사항 정의서 | | |
| 구분 | 요구사항명 | 요구사항 내용 |
| 회원 관련 | Login 로그인 | 회원 로그인 기능 수행 |
| Id\_join 회원가입 | 회원가입 기능 수행 |
| Check\_id 중복조회(id) | Id 중복 조회 기능 수행 |
| Check\_nickname중복조회(nick) | 닉네임 중복 조회 기능 수행 |
| getNickname 닉네임 | 닉네임 불러오기 |
| 포스트 관련 | Insert 등록 | 포스트 등록 기능 수행 |
| SelectAll 목록 | 포스트 전체 목록 조회 기능 수행 |
| Select 확인 | 포스트 확인 기능 수행 |
| Update 수정 | 포스트 수정 기능 수행 |
| Delete 삭제 | 포스트 삭제 기능 수행 |
| parseDate 날짜 | 날짜형을 문자형으로 변환 |
| 댓글 관련 | Insert\_C 등록 | 댓글 등록 기능 수행 |
| SelectAll\_C목록 | 댓글 목록 조회 수행 |

**1. 요구사항(Facebook 프로그램)**

**2. DB 모델링(Facebook 프로그램)**

|  |  |  |  |
| --- | --- | --- | --- |
| Facebook\_MB | | | |
| 컬럼 인덱스 | 컬럼명 | 자료형 | 제약조건 |
| 1 | Id | varchar2(50) | Primary key |
| 2 | Pw | varchar2(50) | Not null |
| 3 | Nickname | varchar2(50) | Unique / not null |
| Post | | | |
| 컬럼 인덱스 | 컬럼명 | 자료형 | 제약조건 |
| 1 | No(글번호) | Number | Primary key |
| 2 | Title(타이틀) | varchar2(100) | Not null |
| 3 | Nickname(닉넴) | varchar2(500) | Foreign key |
| 4. | Register\_date(등록일) | Date |  |
| 5 | Memo(내용) | varchar2(100) | Not null |

|  |  |  |  |
| --- | --- | --- | --- |
| Post\_com | | | |
| 컬럼 인덱스 | 컬럼명 | 자료형 | 제약조건 |
| 1 | Com(댓글내용) | varchar2(50) | Not null |
| 2 | Nickname (닉넴) | varchar2(50) | Not null |
| 3 | Reporing\_date(작성일) | Date |  |
| 4. | No(글번호) | Number | Foreign key  on delete cascade |

|  |
| --- |
| Seq\_pno |
| start with 1  increment by 1  nomaxvalue  nocycle  nocache; |

**전체 코드**

|  |
| --- |
| **DB** |
| create table Facebook\_MB(  id varchar2(50) primary key,  pw varchar2(50) not null,  Nickname varchar2(50) unique not null  );  create table Post(  no number primary key,  title varchar2(30) not null,  nickname varchar2(50) references Facebook\_MB(nickname),  register\_date date,  memo varchar2(300) not null  );  create table Post\_com(  no number references Post(no) on delete cascade,  com varchar2(50) not null,  nickname varchar2(50) not null,  reporing\_date date  );  create sequence seq\_pno  start with 1  increment by 1  nomaxvalue  nocycle  nocache; |

|  |
| --- |
| **FacebookDTO** |
| package com.kh.fb;  public class FacebookDTO {  private String Id;  private String Pw;  private String Nickname;  public FacebookDTO() {}  public FacebookDTO(String id, String pw, String nickname) {  super();  Id = id;  Pw = pw;  Nickname = nickname;  }  public String getId() {  return Id;  }  public void setId(String id) {  Id = id;  }  public String getPw() {  return Pw;  }  public void setPw(String pw) {  Pw = pw;  }  public String getNickname() {  return Nickname;  }  public void setNickname(String nickname) {  Nickname = nickname;  }  } |

|  |
| --- |
| **PostDTO** |
| package com.kh.fb;  public class PostDTO {  private int No;  private String Nickname;  private String Title;  private String Register\_date;  private String Memo;    public PostDTO() {}  public PostDTO(int no, String nickname, String title, String register\_date, String memo) {  super();  No = no;  Nickname = nickname;  Title = title;  Register\_date = register\_date;  Memo = memo;  }  public int getNo() {  return No;  }  public void setNo(int no) {  No = no;  }  public String getNickname() {  return Nickname;  }  public void setNickname(String nickname) {  Nickname = nickname;  }  public String getTitle() {  return Title;  }  public void setTitle(String title) {  Title = title;  }  public String getRegister\_date() {  return Register\_date;  }  public void setRegister\_date(String register\_date) {  Register\_date = register\_date;  }  public String getMemo() {  return Memo;  }  public void setMemo(String memo) {  Memo = memo;  }  @Override  public String toString() {  return No + "\t" + Title + "\t" + Nickname + "\t" + Register\_date;  }  } |

|  |
| --- |
| **CommentDTO** |
| package com.kh.fb;  public class CommentDTO {  private int no;  private String Com;  private String nickname;  private String Reporing\_date;  public CommentDTO() {  }  public CommentDTO(int no, String com, String nickname, String reporing\_date) {  this.no = no;  Com = com;  this.nickname = nickname;  Reporing\_date = reporing\_date;  }  public int getNo() {  return no;  }  public void setNo(int no) {  this.no = no;  }  public String getCom() {  return Com;  }  public void setCom(String com) {  Com = com;  }  public String getNickname() {  return nickname;  }  public void setNickname(String nickname) {  this.nickname = nickname;  }  public String getReporing\_date() {  return Reporing\_date;  }  public void setReporing\_date(String reporing\_date) {  Reporing\_date = reporing\_date;  }  @Override  public String toString() {  return no + "\t" + Com + "\t" + nickname + "\t" + Reporing\_date;  }  } |

|  |
| --- |
| **FacebookDAO** |
| package com.kh.fb;  import java.sql.Connection;  import java.sql.Date;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.text.SimpleDateFormat;  import java.util.ArrayList;  import org.apache.commons.dbcp2.BasicDataSource;  public class FacebookDAO {  BasicDataSource bds = new BasicDataSource();  private static FacebookDAO instance = null;  public FacebookDAO() {  bds.setUrl("jdbc:oracle:thin:@localhost:1521:xe");  bds.setUsername("kh");  bds.setPassword("kh");  bds.setInitialSize(10);  }  public static FacebookDAO getInstance() {  if (instance == null) {  instance = new FacebookDAO();  }  return instance;  }  public Connection getConnection() throws Exception {  return bds.getConnection();  }  // 로그인  public boolean Login(String Id, String Pw) throws Exception {  String sql = "select \* from Facebook\_MB where Id = ? and Pw = ?";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql)) {  pstmt.setString(1, Id);  pstmt.setString(2, Pw);  ResultSet rs = pstmt.executeQuery();  if (rs.next()) {  return true;  }  return false;  }  }  // 닉네임 불러오기  public String getNickname(String Id) throws Exception {  String sql = "select nickname from Facebook\_MB where Id = ?";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql)) {  pstmt.setString(1, Id);  ResultSet rs = pstmt.executeQuery();  if (rs.next()) {  return rs.getString(1);  }  return null;  }  }  // 회원가입  public int Id\_join(FacebookDTO dto) throws Exception {  String sql = "insert into Facebook\_MB values(?,?,?)";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setString(1, dto.getId());  pstmt.setString(2, dto.getPw());  pstmt.setString(3, dto.getNickname());  int rs = pstmt.executeUpdate();  return rs;  }  }  // 닉네임 중복체크  public boolean check\_nickname(String nickname) throws Exception {  String sql = "select \* from Facebook\_MB where nickname = ?";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setString(1, nickname);  ResultSet rs = pstmt.executeQuery();  if (rs.next()) { // 중복된 닉네임이 있음  return false; // 사용불가 닉네임 false 리턴  }  return true; // 사용할 수 있는 닉네임일 경우 true 리턴  }  }  // 아이디중복체크  public boolean check\_id(String id) throws Exception {  String sql = "select \* from Facebook\_MB where id = ?";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setString(1, id);  ResultSet rs = pstmt.executeQuery();  if (rs.next()) { // 중복된 아이디가 있음  return false; // 사용불가 아이디 false 리턴  }  return true; // 사용할 수 있는 아이디일 경우 true 리턴  }  }  // insert  public int insert(PostDTO dto) throws Exception {  String sql = "insert into Post values(seq\_pno.nextval,?,?,sysdate,?)";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setString(1, dto.getTitle());  pstmt.setString(2, dto.getNickname());  pstmt.setString(3, dto.getMemo());  int rs = pstmt.executeUpdate();  return rs;  }  }  // 포스트 목록 다 출력  public ArrayList<PostDTO> selectAll() throws Exception {  String sql = "select \* from Post";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  ResultSet rs = pstmt.executeQuery();  ArrayList<PostDTO> list = new ArrayList<>();  while (rs.next()) {  int no = rs.getInt(1);  String title = rs.getString(2);  String nickname = rs.getString(3);  String register\_date = parseDate(rs.getDate(4));  String memo = rs.getString(5);  list.add(new PostDTO(no, title, nickname, register\_date, memo));  }  return list;  }  }  // 번호받아서 하나 출력  public PostDTO select(int pno) throws Exception {  String sql = "select \* from Post where no = ?";  try (Connection con = getConnection(); PreparedStatement pstmt = con.prepareStatement(sql)) {  pstmt.setInt(1, pno);  ResultSet rs = pstmt.executeQuery();  if (rs.next()) {  int no = rs.getInt(1);  String nickname = rs.getString(3);  String register\_date = parseDate(rs.getDate(4));  String title = rs.getString(2);  String memo = rs.getString(5);  PostDTO dto = new PostDTO(no, nickname, title, register\_date, memo);  return dto;  }  return null;  }  }  // 업데이트  public int update(PostDTO dto) throws Exception {  String sql = "update Post set title = ?, memo = ? where no = ?";  try (Connection con = getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setString(1, dto.getTitle());  pstmt.setString(2, dto.getMemo());  pstmt.setInt(3, dto.getNo());  int rs = pstmt.executeUpdate();  return rs;  }  }  public int insert\_C(int no, String com, String nickname) throws Exception { // 덧글 추가  String sql = "insert into post\_com values(?,?,?,sysdate)";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setInt(1, no);  pstmt.setString(2, com);  pstmt.setString(3, nickname);  int rs = pstmt.executeUpdate();  return rs;  }  }  public ArrayList<CommentDTO> selectAll\_c(int num) throws Exception { // 댓글 목록  String sql = "select \* from post\_com where no = ?";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setInt(1, num);  ResultSet rs = pstmt.executeQuery();  ArrayList<CommentDTO> list = new ArrayList<>();  while (rs.next()) {  String Com = rs.getString(2);  String Nickname = rs.getString(3);  String Reporing\_date = parseDate(rs.getDate(4));  int no = rs.getInt(1);  list.add(new CommentDTO(no, Com, Nickname, Reporing\_date));  }  return list;  }  }  public int delete(int no) throws Exception { // 포스트 삭제  String sql = "delete from post where no= ? ";  try (Connection con = this.getConnection(); PreparedStatement pstmt = con.prepareStatement(sql);) {  pstmt.setInt(1, no);  // pstmt.setString(2, nickname);  int rs = pstmt.executeUpdate();  return rs;  }  }  public String parseDate(Date date) {  SimpleDateFormat sdf = new SimpleDateFormat("yyyy년 MM월 dd일 hh:mm");  return sdf.format(date);  }  } |

|  |
| --- |
| **Run** |
| package com.kh.fb;  import java.util.ArrayList;  import java.util.Scanner;  public class Run {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  FacebookDAO dao = FacebookDAO.getInstance();  login: while (true) {  System.out.println("Facebook 프로그램");  System.out.println("1. 로그인");  System.out.println("2. 회원가입");  System.out.println("0. 프로그램 종료");  System.out.print(">> ");  int menu = Integer.parseInt(sc.nextLine());  if (menu == 1) { // 로그인  System.out.print("id 입력 >> ");  String Id = sc.nextLine();  System.out.print("pw 입력 >> ");  String Pw = sc.nextLine();  try {  if (dao.Login(Id, Pw)) {  String Nickname = dao.getNickname(Id);  System.out.println("로그인 성공");  System.out.println("\*\*\* " + Nickname + "님 환영합니다 \*\*\*\n");  while (true) {  System.out.println("1. 포스트 등록");  System.out.println("2. 포스트 목록");  System.out.println("3. 포스트 확인");  System.out.println("4. 포스트 수정");  System.out.println("5. 포스트 삭제");  System.out.println("6. 로그아웃");  System.out.print(">> ");  int pmenu = Integer.parseInt(sc.nextLine());  if (pmenu == 1) { // insert  try {  System.out.println("=== 포스트 등록 ===");  while (true) {  System.out.print("타이틀(10자 이내) >> ");  String title = sc.nextLine();  if (title.length() > 10) {  System.out.println("타이틀은 10자 이내로 입력하세요.");  } else {  System.out.print("내용(100자 이내) >> ");  String memo = sc.nextLine();  if (memo.length() > 100) {  System.out.println("내용은 100자 이내로 입력하세요.");  } else {  PostDTO dto = new PostDTO(0, Nickname, title, null, memo);  int rs = dao.insert(dto);  if (rs > 0) {  System.out.println("포스트 등록 완료\n");  } else {  System.out.println("포스트 등록 실패\n");  }  break;  }  }  }  } catch (Exception e) {  e.printStackTrace();  System.out.println("너무 길어");  }  } else if (pmenu == 2) { // selectAll 목록  System.out.println("==== 포스트 목록==== ");  System.out.println("글번호 \t타이틀 \t닉네임 \t등록일");  try {  ArrayList<PostDTO> list = dao.selectAll();  if (list != null) {  for (PostDTO dto : list) {  System.out.println(dto.toString());  System.out.println("\n");  }  } else {  System.out.println("포스트 목록을 불러오지 못했습니다.");  }  } catch (Exception e) {  e.printStackTrace();  System.out.println("오류");  }  } else if (pmenu == 3) { // select 확인  ArrayList<PostDTO> list = dao.selectAll();  if (list != null) {  for (PostDTO dto : list) {  System.out.println(dto.toString());  }  } else {  System.out.println("포스트 목록을 불러오지 못했습니다.");  }  while (true) {  System.out.print("확인할 포스트의 번호를 입력하세요 >> ");  int num = Integer.parseInt(sc.nextLine());  PostDTO dto = dao.select(num);  System.out.println("=== 포스트 확인 ===");  System.out.println("글번호\t" + dto.getNo());  System.out.println("닉네임\t" + dto.getNickname());  System.out.println("등록일\t" + dto.getTitle());  System.out.println("내용\t" + dto.getMemo());  try {// selectAll\_C  ArrayList<CommentDTO> list2 = dao.selectAll\_c(num);  if (list != null) {  for (CommentDTO cdto : list2) {  System.out.println(cdto.toString());  }  } else {  System.out.println("데이터 조회 실패");  }  } catch (Exception e) {  e.printStackTrace();  }  System.out.println("\n1. 댓글 등록");  System.out.println("2. 메뉴로 돌아가기");  System.out.print(">> ");  int menu\_c = Integer.parseInt(sc.nextLine());  if (menu\_c == 1) {  System.out.print("댓글 입력(50자 이내) >> ");  String com = sc.nextLine();  int rs = dao.insert\_C(num, com, Nickname);  if (rs > 0) {  System.out.println("댓글이 등록되었습니다.\n");  } else {  System.out.println("댓글 등록이 실패되었습니다.\n");  }  } else if (menu\_c == 2) {  break;  }  }  // 댓글등록,메뉴로돌아가기  } else if (pmenu == 4) { // update 수정  // 포스트목록  System.out.println("=== 포스트 목록 ===");  // 포스트목록불러오기  System.out.print("수정할 포스트의 번호를 입력하세요 >> ");  int no = Integer.parseInt(sc.nextLine());  System.out.print("수정할 타이틀(10자 이내) >> ");  String title = sc.nextLine();  System.out.print("수정할 내용(100자 이내) >> ");  String memo = sc.nextLine();  PostDTO dto2 = new PostDTO(no, null, title, null, memo);  int rs = dao.update(dto2);  if (rs > 0) {  System.out.println("포스트 수정 완료\n");  }  } else if (pmenu == 5) { // delete 삭제  System.out.print("삭제할 포스트의 번호를 입력하세요 >> ");  int no = Integer.parseInt(sc.nextLine());  System.out.print("정말 삭제하시겠습니까 ? (Y/N) >> ");  String select = sc.nextLine();  if (select.equals("Y")) {  try {  int rs = dao.delete(no);  if (rs > 0) {  System.out.println("삭제가 완료되었습니다.\n");  }  } catch (Exception e) {  e.printStackTrace();  System.out.println("DB 접속이 불안정합니다.");  }  } else if (select.equals("N")) {  System.out.println("삭제가 취소 되었습니다.\n");  }  } else if (pmenu == 6) {// 로그아웃  System.out.println("\n");  break;  }  }  }  } catch (Exception e) {  e.printStackTrace();  }  } else if (menu == 2) { // 회원가입  String Id = null;  String Nickname = null;  while (true) {  System.out.print("id 입력 >> ");  Id = sc.nextLine();  try {  if (dao.check\_id(Id)) {  System.out.println("사용가능한 아이디입니다.");  break;  } else {  System.out.println("존재하는 아이디입니다. 다른 아이디를 사용하세요");  continue;  }  } catch (Exception e) {  e.printStackTrace();  }  }  System.out.print("pw 입력 >> ");  String Pw = sc.nextLine();  while (true) {  System.out.print("nickname 입력 >> ");  Nickname = sc.nextLine();  try {  if (dao.check\_nickname(Nickname)) {  System.out.println("사용가능한 닉네임입니다.");  break;  } else {  System.out.println("존재하는 닉네임입니다. 다른 닉네임을 사용하세요");  continue;  }  } catch (Exception e) {  e.printStackTrace();  }  }  try {  int rs = dao.Id\_join(new FacebookDTO(Id, Pw, Nickname));  if (rs > 0) {  System.out.println("회원가입에 성공하였습니다.\n");  continue;  } else {  System.out.println("회원가입에 실패하였습니다.\n");  }  } catch (Exception e) {  e.printStackTrace();  System.out.println("오류가 발생하였습니다.\n");  }  break;  } else if (menu == 0) { // 프로그램 종료  System.out.println("프로그램을 종료합니다.");  break login;  } else {  System.out.println("번호를 다시 입력하세요");  }  }  }  } |