L O N U A

(All For Individual Customized Fashion)

Team D.P(Developer_Passion)

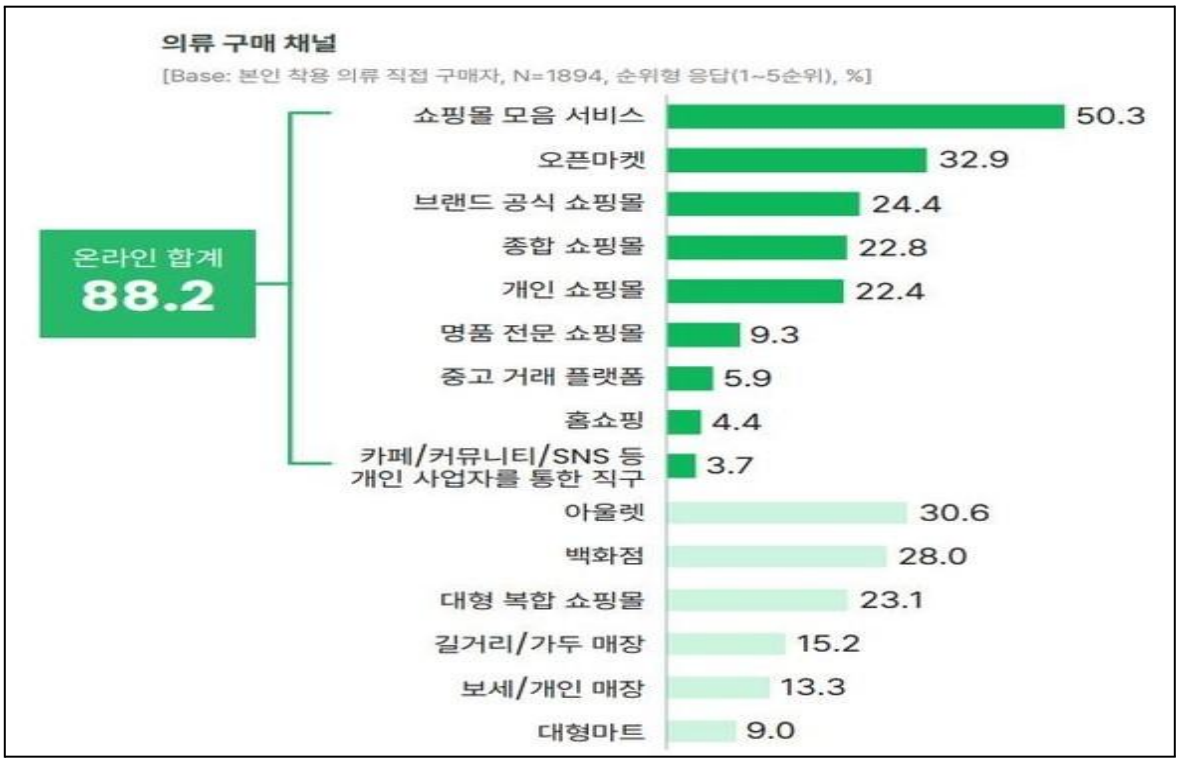
목 차

프로젝트 개요

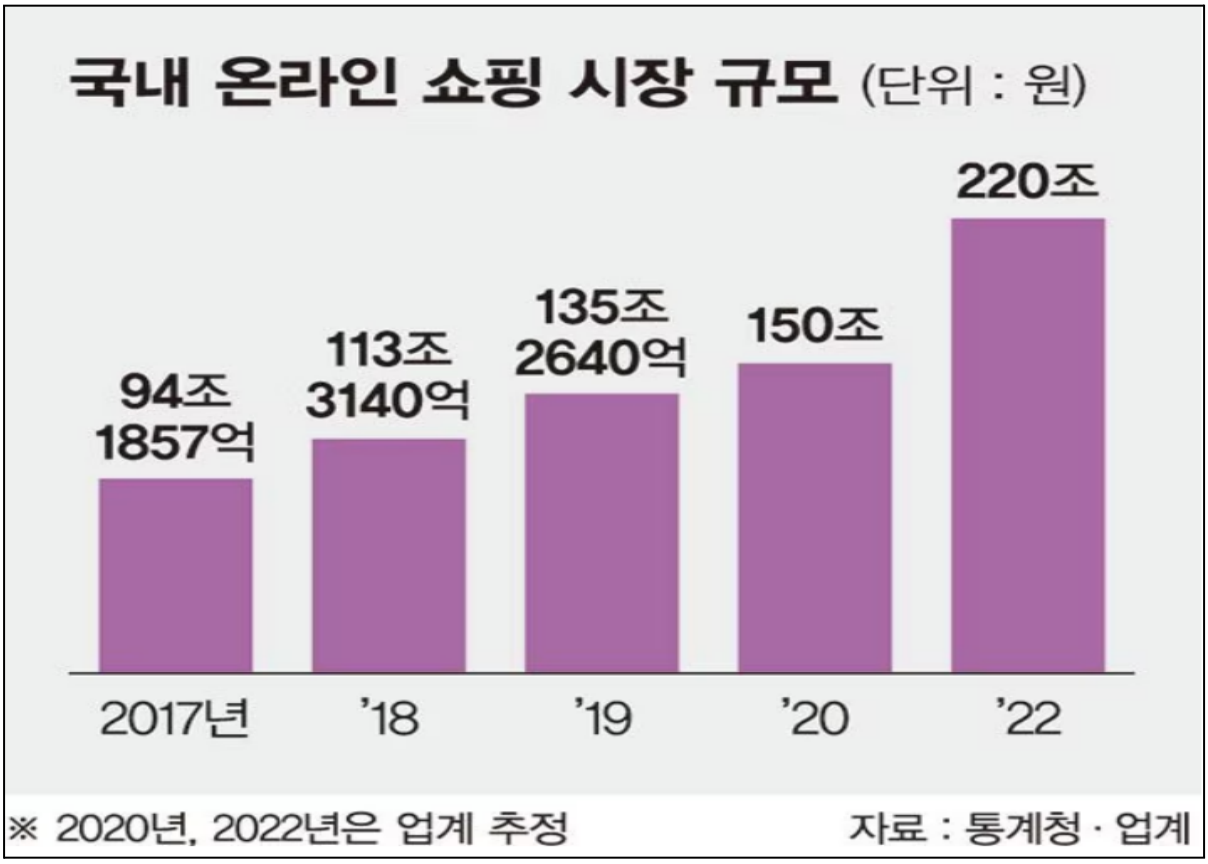
1.1 소 개

"LONUA(All For Individual Customized Fashion)" 는 개인의 신체 치수, 선호하는 스타일 등을 회원가입 시 입력함으로써 수많은 상품에 대한 이용자의 선택의 폭을 줄임으로써 쇼핑시간을 단축 시켜주는 **"개인 맞춤형 패션 플랫폼 서비스"** 를 제공하는 쇼핑몰이다.

1.2 배 경



소비자 데이터 플랫폼 "오픈서베이" 에 따르면 의류 구매 채널에 대해 만 15세 ~ 39세 남녀 2천명을 대상으로 조사한 결과, 온라인 쇼핑몰을 이용하는 사람의 수가 88.2% 로 10명 가운데 9명은 온라인 상에서 의류를 구매하는 것으로 나타났다.



통계청의 조사결과에 따르면, 국내 온라인 쇼핑 시장 규모가 2020년 150조 대비하여 2022년에는 70조가 증가한 220조로 2년 사이 약 69%가 성장한 것으로 나타났다.

이처럼, 온라인 쇼핑몰 이용자 수가 **"지속적으로 증가"** 하고 있는 만큼, 쇼핑몰에 등록되는 상품의 수 역시 **"기하급수적으로 늘어나"** 상품 선택 시 이용자가 선택의 어려움을 겪고 있는 현실이다.

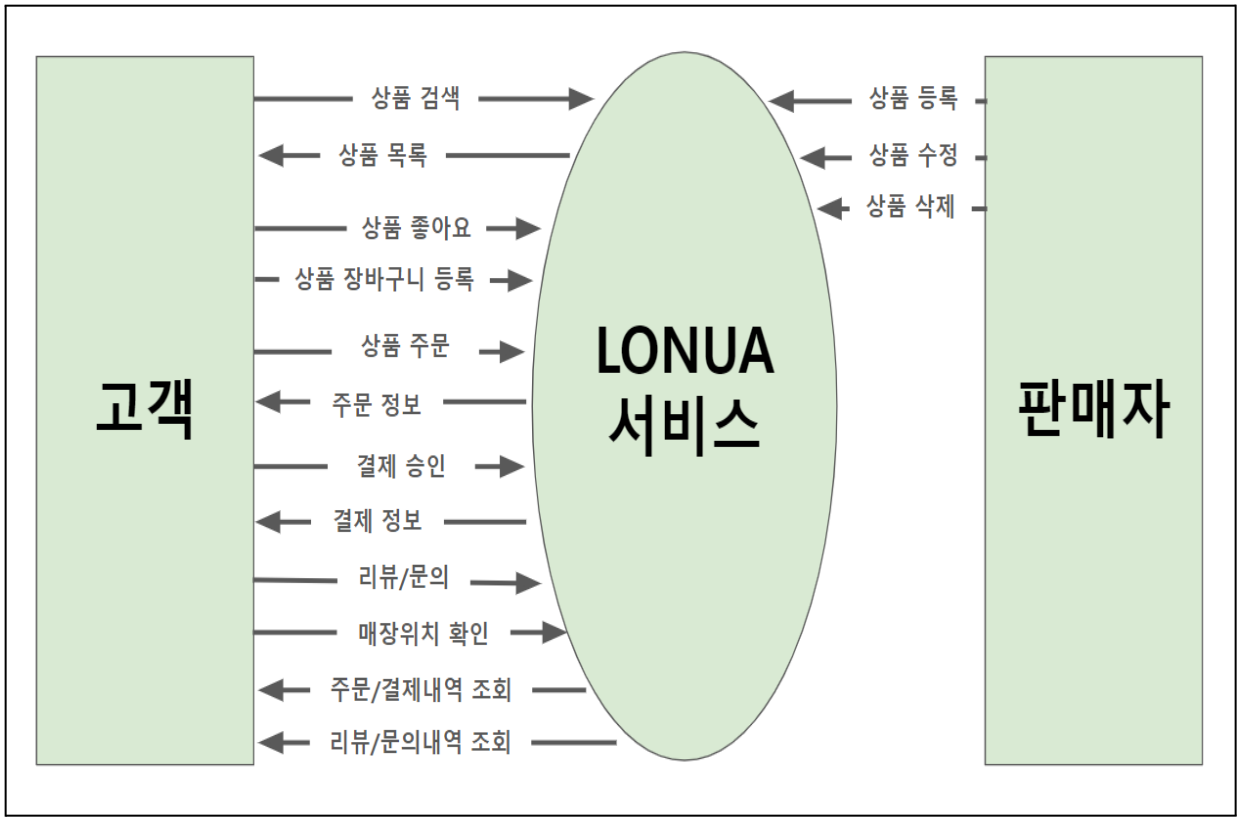
따라서 개인에게 맞는 옷, 스타일을 제공하여 수많은 상품에 대한 이용자의 선택의 폭을 줄이고, 쇼핑시간을 단축 시킴으로써 이용자가 쇼핑 시 선택의 어려움을 해결해주기 위해 **"LONUA"** 프로젝트를 계획하였다.

1.3 시나리오

"LONUA" 쇼핑몰은 아래와 같이 11가지 핵심 기능을 가지고 있다.

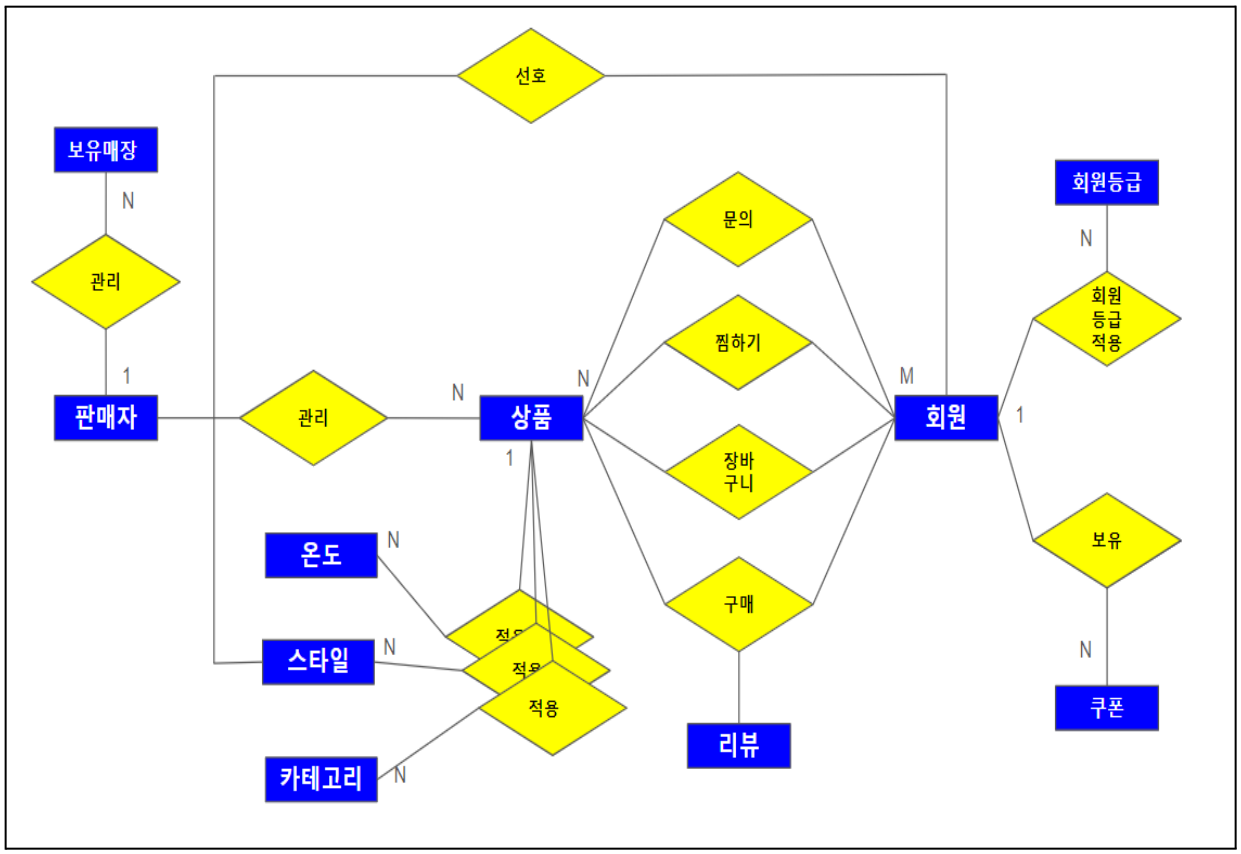
1. 회원은 ID, PW, email, 신체치수, 선호 스타일 등을 입력하여 회원가입이 가능하다.
2. 회원은 회원가입 시 입력한 ID 와 PW를 통해 로그인이 가능하다.
3. 회원은 마음에 드는 상품을 장바구니에 담거나 찜할 수 있다.
4. 회원은 원하는 상품을 주문 및 결제 할 수 있다.
5. 회원은 원하는 온도와, 스타일, 카테고리 별로 상품을 검색할 수 있다.
6. 회원은 구매한 상품에 대하여 리뷰(평점 포함)를 남길 수 있다.
7. 회원은 원하는 주소를 입력하면 해당 주소 근처 쇼핑몰에 입점한 브랜드의 매장 위치를 확인 할 수 있다.
8. 회원은 상품에 대한 문의 글을 남길 수 있다.
9. 회원은 지급받은 쿠폰을 사용할 수 있다.
10. 회원은 월별 구매한 금액에 따라 회원등급이 부여되고 등급에 따라 할인율을 적용받을 수 있다.
11. 판매자는 판매할 상품을 관리(등록, 삭제 등) 할 수 있다.

업무 배경도


고객
상품 검색
상품 목록
상품 좋아요
상품 장바구니 등록
상품 주문
주문 정보
결제 승인
결제 정보
리뷰/문의
매장위치 확인
주문/결제내역 조회
리뷰/문의내역 조회
LONUA
서비스
상품 등록
상품 수정
상품 삭제
판매자

설 계 (E-R 다이어그램)

3.1 ERD (관 계)



3.2 테이블 별 속성

테이블	속 성
판매자	판매자IDX, 브랜드 ID, 브랜드 PW, 브랜드 소개, 브랜드 사진, 연락처, 브랜드 스타일, 영업 소재지, 생성 날짜, 수정 날짜, 사업자 등록증, 통신판매업 신고증, 통장 번호, 반품 주소, 반품 비용, 반품 택배사, 상태
회원	회원IDX, ID, PW, 이름, 생년월일, 성별, 휴대전화번호, 휴대전화인증, 집 주소, email, email인증, PW 변경 일시, 상태, 소셜계정가입여부 선호스타일1, 선호스타일2, 선호스타일3, 하의 총길이, 엉덩이 둘레, 밑단, 허벅지 둘레, 허리 둘레, 가슴 둘레, 밑위, 상의 총길이, 팔길이, 어깨넓이, 회원 등급, 등급 수정 날짜, 마일리지, 상태(등록/휴먼, 탈퇴)
회원등급	등급IDX, 등급 종류, 할인율, 등업조건
스타일	스타일IDX, 스타일 이름, 스타일 설명
상품	상품IDX, 브랜드ID, 상품 이름, 수량, 가격, 옵션(색깔), 생성 날짜, 수정 날짜 어깨 넓이, 가슴 둘레, 팔 길이, 상의 총 길이, 엉덩이 둘레, 허벅지 둘레, 밑위, 허리둘레, 밑단, 스타일1_IDX, 스타일2_IDX, 스타일3_IDX, 카테고리,적정 온도1_IDX, 적정 온도2_IDX, 적정 온도3_IDX 상태(등록/삭제)
카테고리	카테고리IDX, 설명, 상태
적정온도	온도IDX, 온도
쿠폰	쿠폰IDX, 쿠폰명, 쿠폰 설명, 쿠폰 할인율, 쿠폰 유효기간, 쿠폰 받은 날짜, 쿠폰 사용 마감일
장바구니	회원IDX, 상품IDX, 생성 날짜, 수정 날짜, 상태(등록/삭제)
주문내역	주문IDX, 회원IDX, 상품IDX, 생성 날짜, 수정 날짜, 상태 (입금대기 / 입금완료 / 배송 전 / 배송 중 / 배송 완료)
찜하기	좋아요IDX, 회원IDX, 상품IDX, 생성 날짜, 수정 날짜, 상태(등록/삭제)
리뷰	리뷰IDX, 주문IDX, 리뷰 내용, 리뷰 사진, 별점, 생성 날짜, 수정 날짜, 상태(등록/삭제)
문의	문의IDX, 회원IDX, 상품IDX, 문의 종류(상품문의/배송문의/사이즈문의), 문의 내용, 답변여부(Y/N), 비밀여부(Y/N)

구 현

4.1 릴레이션

테이블	**SQL** 쿼리문
판매자	CREATE TABLE IF NOT EXISTS `lonua`.`brand` (`brand_idx` INT NOT NULL AUTO_INCREMENT, `brand_id` VARCHAR(45) NOT NULL, `brand_pw` VARCHAR(45) NOT NULL, `brand_photo` VARCHAR(45) NULL DEFAULT NULL, `brand_introduction` VARCHAR(500) NULL DEFAULT NULL, `brand_style` INT NULL DEFAULT NULL, `phone_number` VARCHAR(20) NULL DEFAULT NULL, `business_registration` VARCHAR(45) NULL DEFAULT NULL, `mail_order_business_report_card` VARCHAR(45) NULL DEFAULT NULL, `bank_account_number` VARCHAR(45) NULL DEFAULT NULL, `return_address` VARCHAR(45) NULL DEFAULT NULL, `return_cost` INT NULL DEFAULT NULL, `return_courier` VARCHAR(45) NULL DEFAULT NULL, `business_address` VARCHAR(45) NULL DEFAULT NULL, `creation_date` TIMESTAMP NULL DEFAULT NULL, `modification_date` TIMESTAMP NULL DEFAULT NULL, `status` TINYINT NULL DEFAULT NULL, PRIMARY KEY (`brand_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;

테이블	**SQL** 쿼리문
회 원	CREATE TABLE IF NOT EXISTS `lonua`.`user` (`user_idx` INT NOT NULL AUTO_INCREMENT, `user_name` VARCHAR(45) NULL DEFAULT NULL, `user_birth` VARCHAR(45) NULL DEFAULT NULL, `user_gender` VARCHAR(45) NULL DEFAULT NULL, `user_phone` VARCHAR(45) NULL DEFAULT NULL, `user_addr` VARCHAR(45) NULL DEFAULT NULL, `prefer_style` INT NULL DEFAULT NULL, `shoulder_width` INT NULL DEFAULT NULL, `chest_size` INT NULL DEFAULT NULL, `arm_length` INT NULL DEFAULT NULL, `top_length` INT NULL DEFAULT NULL, `waistline` INT NULL DEFAULT NULL, `hip_circumference` INT NULL DEFAULT NULL, `thigh_circumference` INT NULL DEFAULT NULL, `crotch_length` INT NULL DEFAULT NULL, `hem_length` INT NULL DEFAULT NULL, `total_bottom_lenght` INT NULL DEFAULT NULL, `mileage` INT NULL DEFAULT NULL, `update_date` TIMESTAMP NULL DEFAULT NULL, `status` INT NULL DEFAULT NULL, `user_class` INT NOT NULL, `user_id` VARCHAR(45) NULL DEFAULT NULL, `user_email` VARCHAR(45) NULL DEFAULT NULL, `email_authen_status` INT NULL DEFAULT NULL, `phone_authen_status` INT NULL DEFAULT NULL, `pw_changedate` TIMESTAMP NULL DEFAULT NULL, `social_account` VARCHAR(45) NULL DEFAULT NULL, PRIMARY KEY (`user_idx`), INDEX `fk_user_info_user_class1_idx` (`user_class` ASC) VISIBLE, CONSTRAINT `fk_user_info_user_class1` FOREIGN KEY (`user_class`) REFERENCES `lonua`.`user_class` (`class_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;

테이블	**SQL** 쿼리문
회원등급	CREATE TABLE IF NOT EXISTS `lonua`.`user_class` (`class_idx` INT NOT NULL AUTO_INCREMENT, `class_type` VARCHAR(45) NULL DEFAULT NULL, `discount_rate` INT NULL DEFAULT NULL, `class_up_condition` INT NULL DEFAULT NULL, PRIMARY KEY (`class_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;
스타일	CREATE TABLE IF NOT EXISTS `lonua`.`style` (`style_idx` INT NOT NULL AUTO_INCREMENT, `name` VARCHAR(45) NULL DEFAULT NULL, `description` VARCHAR(100) NULL DEFAULT 'style', PRIMARY KEY (`style_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;
카테고리	CREATE TABLE IF NOT EXISTS `lonua`.`category` (`category_idx` INT NOT NULL AUTO_INCREMENT, `description` VARCHAR(100) NOT NULL, `status` INT NOT NULL DEFAULT '0', PRIMARY KEY (`category_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;
적정온도	CREATE TABLE IF NOT EXISTS `lonua`.`proper_temperature` (`temperature_idx` INT NOT NULL AUTO_INCREMENT, `temperature` INT NULL DEFAULT NULL, PRIMARY KEY (`temperature_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;

테이블	**SQL** 쿼리문

상 품	CREATE TABLE IF NOT EXISTS `lonua`.`product` (
	`product_idx` INT NOT NULL AUTO_INCREMENT, `brand_id` INT NOT NULL,
	`product_name` VARCHAR(100) NOT NULL, `quantity` INT NOT NULL,
	`price` INT NOT NULL, `product_option` VARCHAR(100) NULL DEFAULT NULL,
	`shoulder_width` INT NULL DEFAULT NULL, `chest_size` INT NULL DEFAULT
	NULL, `arm_length` INT NULL DEFAULT NULL, `top_length` INT NULL DEFAULT
	NULL, `waistline` INT NULL DEFAULT NULL, `hip_circumference` INT NULL
	DEFAULT NULL, `thigh_circumference` INT NULL DEFAULT NULL,
	`crotch_length` INT NULL DEFAULT NULL, `hem_length` INT NULL DEFAULT
	NULL, `total_bottom_length` INT NULL DEFAULT NULL, `category_idx` INT
	NULL DEFAULT NULL,`creation_date` TIMESTAMP NULL DEFAULT NULL,
	`modification_date` TIMESTAMP NULL DEFAULT NULL,`status` INT NULL
	DEFAULT '0', `category_id` INT NULL DEFAULT NULL, `proper_temperature1`
	INT NULL DEFAULT NULL, `proper_temperature2` INT NULL DEFAULT NULL,
	`proper_temperature3` INT NULL DEFAULT NULL, `style1` INT NULL DEFAULT
	NULL, `style2` INT NULL DEFAULT NULL, `style3` INT NULL DEFAULT NULL,
	PRIMARY KEY (`product_idx`), INDEX `fk_product_category1_idx` (`category_id`
	ASC) VISIBLE, INDEX `fk_product_style1_idx` (`style1` ASC) VISIBLE, INDEX
	`fk_product_proper_temperature1_idx` (`proper_temperature1` ASC) VISIBLE,
	INDEX `fk_product_proper_temperature2_idx` (`proper_temperature2` ASC)
	VISIBLE, INDEX `fk_product_proper_temperature3_idx` (`proper_temperature3`
	ASC) VISIBLE, INDEX `fk_product_style2_idx` (`style2` ASC) VISIBLE, INDEX
	`fk_product_style3_idx` (`style3` ASC) VISIBLE, INDEX `fk_product_brand1_idx`
	(`brand_id` ASC) VISIBLE, CONSTRAINT `fk_product_brand1` FOREIGN KEY
	(`brand_id`) REFERENCES `lonua`.`brand` (`brand_idx`), CONSTRAINT
	`fk_product_category1` FOREIGN KEY (`category_id`) REFERENCES
	`lonua`.`category` (`category_idx`), CONSTRAINT `fk_product_style1`
	FOREIGN KEY (`style1`) REFERENCES `lonua`.`style` (`style_idx`), CONSTRAINT
	`fk_product_style2` FOREIGN KEY (`style2`) REFERENCES `lonua`.`style`
	(`style_idx`), CONSTRAINT `fk_product_style3` FOREIGN KEY (`style3`)
	REFERENCES `lonua`.`style` (`style_idx`), CONSTRAINT `proper_temperature1`
	FOREIGN KEY (`proper_temperature1`) REFERENCES
	`lonua`.`proper_temperature` (`temperature_idx`), CONSTRAINT
	`proper_temperature2` FOREIGN KEY (`proper_temperature2`)
	REFERENCES `lonua`.`proper_temperature` (`temperature_idx`), CONSTRAINT
	`proper_temperature3` FOREIGN KEY (`proper_temperature3`) REFERENCES
	`lonua`.`proper_temperature` (`temperature_idx`))
	ENGINE = InnoDB
	AUTO_INCREMENT = 26
	DEFAULT CHARACTER SET = utf8mb4
	COLLATE = utf8mb4_0900_ai_ci;

테이블	**SQL** 쿼리문
쿠 폰	CREATE TABLE IF NOT EXISTS `lonua`.`coupon` (`coupon_idx` INT NOT NULL AUTO_INCREMENT, `coupon_name` VARCHAR(45) NULL DEFAULT NULL, `coupon_description` VARCHAR(45) NULL DEFAULT NULL, `coupon_discount_rate` INT NULL DEFAULT NULL, `coupon_expiration_date` TIMESTAMP NULL DEFAULT NULL, `received_date` TIMESTAMP NULL DEFAULT NULL, `usedeadline` TIMESTAMP NULL DEFAULT NULL, `status` INT NULL DEFAULT NULL, `user_id` INT NOT NULL, PRIMARY KEY (`coupon_idx`, `user_id`), INDEX `fk_coupon_user_info1_idx` (`user_id` ASC) VISIBLE, CONSTRAINT `fk_coupon_user_info1` FOREIGN KEY (`user_id`) REFERENCES `lonua`.`user` (`user_idx`)) ENGINE = InnoDB AUTO_INCREMENT = 21 DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;
장바구니	CREATE TABLE IF NOT EXISTS `lonua`.`cart` (`creation_date` TIMESTAMP NOT NULL, `modification_date` TIMESTAMP NOT NULL, `status` INT NOT NULL, `product_id` INT NOT NULL, `user_id` INT NOT NULL, PRIMARY KEY (`product_id`, `user_id`), INDEX `fk_cart_product11_idx` (`product_id` ASC) VISIBLE, INDEX `fk_cart_user_info11_idx` (`user_id` ASC) VISIBLE, CONSTRAINT `fk_cart_product11` FOREIGN KEY (`product_id`) REFERENCES `lonua`.`product` (`product_idx`), CONSTRAINT `fk_cart_user_info11` FOREIGN KEY (`user_id`) REFERENCES `lonua`.`user` (`user_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;

테이블	**SQL** 쿼리문
주문내역	CREATE TABLE IF NOT EXISTS `lonua`.`order` (`order_idx` INT NOT NULL, `creation_date` TIMESTAMP NOT NULL, `modification_date` TIMESTAMP NOT NULL, `status` INT NOT NULL, `product_id` INT NOT NULL, `user_id` INT NOT NULL, PRIMARY KEY (`order_idx`), INDEX `fk_order_product11_idx` (`product_id` ASC) VISIBLE, INDEX `fk_order_user_info11_idx` (`user_id` ASC) VISIBLE, CONSTRAINT `fk_order_product11` FOREIGN KEY (`product_id`) REFERENCES `lonua`.`product` (`product_idx`), CONSTRAINT `fk_order_user_info11` FOREIGN KEY (`user_id`) REFERENCES `lonua`.`user` (`user_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;
찜하기	CREATE TABLE IF NOT EXISTS `lonua`.`like` (`like_idx` INT NOT NULL, `creation_date` TIMESTAMP NULL DEFAULT NULL, `modification_date` TIMESTAMP NULL DEFAULT NULL, `status` INT NOT NULL, `product_id` INT NOT NULL, `user_id` INT NOT NULL, PRIMARY KEY (`like_idx`), INDEX `fk_like_product11_idx` (`product_id` ASC) VISIBLE, INDEX `fk_like_user_info11_idx` (`user_id` ASC) VISIBLE, CONSTRAINT `fk_like_product11` FOREIGN KEY (`product_id`) REFERENCES `lonua`.`product` (`product_idx`), CONSTRAINT `fk_like_user_info11` FOREIGN KEY (`user_id`) REFERENCES `lonua`.`user` (`user_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;

테이블	**SQL** 쿼리문
리 뷰	CREATE TABLE IF NOT EXISTS `lonua`.`review` (`review_idx` INT NOT NULL, `review_content` VARCHAR(300) NOT NULL, `review_photo` VARCHAR(100) NULL DEFAULT NULL, `evaluation` INT NOT NULL, `creation_date` TIMESTAMP NOT NULL, `modification_date` TIMESTAMP NOT NULL, `status` INT NOT NULL, `order_id` INT NOT NULL, PRIMARY KEY (`review_idx`), INDEX `fk_review_order1_idx` (`order_id` ASC) VISIBLE, CONSTRAINT `fk_review_order1` FOREIGN KEY (`order_id`) REFERENCES `lonua`.`order` (`order_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;
문 의	CREATE TABLE IF NOT EXISTS `lonua`.`question` (`question_idx` INT NOT NULL, `question_type` VARCHAR(10) NOT NULL, `question_content` VARCHAR(300) NOT NULL, `answer_status` VARCHAR(1) NOT NULL, `secret_status` VARCHAR(1) NOT NULL, `product_id` INT NOT NULL, `user_id` INT NOT NULL, PRIMARY KEY (`question_idx`), INDEX `fk_question_product11_idx` (`product_id` ASC) VISIBLE, INDEX `fk_question_user_info11_idx` (`user_id` ASC) VISIBLE, CONSTRAINT `fk_question_product11` FOREIGN KEY (`product_id`) REFERENCES `lonua`.`product` (`product_idx`), CONSTRAINT `fk_question_user_info11` FOREIGN KEY (`user_id`) REFERENCES `lonua`.`user` (`user_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;

4.2 기능별 쿼리문

4.2.1 회원기능

테이블	**SQL** 쿼리문
회원가입	INSERT INTO user(user_idx, user_name, user_birth, user_gender, user_phone, user_addr, prefer_style,shoulder_width, chest_size, arm_length, top_length, waistline, hip_circumference, thigh_circumference,crotch_length, hem_length, total_bottom_lenght, mileage, update_date, status, user_class, user_id,user_email, email_authen_status, phone_authen_status, pw_changedate, social_account) VALUES([회원 idx](INT), [유저 입력_회원이름]('홍길동'), [유저 입력_생년월일](yyyy-mm-dd), [유저 입력_성별](남성일 경우 'm', 여성일 경우 'w'), [유저 입력_전화번호](000-0000-0000), [유저 입력_주소](문자열), [선호스타일id](외래키), [치수](INT), [치수](INT), [치수](INT), [치수](INT), [치수](INT), [치수](INT), [치수](INT), [치수](INT), [치수](INT), [치수](INT), [회원 정보 수정 시간](now() 함수 이용), [유저 상태](활성 계정일 경우: 1, 휴면 계정일 경우: 0), [회원등급_id](외래키), [유저 입력_ID](문자열), [유저 입력_이메일 주소](sample@email.com), [이메일 인증 여부](0 or 1), [핸드폰 인증 여부](0 or 1), [비밀번호 변경 일시](yyyy-mm-dd), [소셜 계정 접속 여부]('Naver'));
로그인 기능	SELECT * FROM user WHERE user_id=[유저 입력_ID] AND user_pw=[유저 입력_PW];
개인정보 변경	UPDATE user SET [속성명]=[유저입력_값] WHERE user_idx=[접속해 있는 유저의 IDX];
매 장	CREATE TABLE IF NOT EXISTS `lonua`.`store` (`store_idx` INT NOT NULL, `store_address` VARCHAR(45) NULL DEFAULT NULL, `brand_brand_idx` INT NOT NULL, PRIMARY KEY (`store_idx`), INDEX `fk_store_brand_idx` (`brand_brand_idx` ASC) VISIBLE, CONSTRAINT `fk_store_brand` FOREIGN KEY (`brand_brand_idx`) REFERENCES `lonua`.`brand` (`brand_idx`)) ENGINE = InnoDB DEFAULT CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci;

4.2.2 장바구니 기능

테이블	**SQL** 쿼리문
원하는 상품 장바구니에 담기	INSERT INTO cart(creation_date, modification_date, status, product_id, user_id)VALUES ([장바구니 추가 시간](now() 내장 함수 사용),[장바구니 변경 시간](now() 내장 함수 사용),[장바구니에 담긴 상품의 상태](장바구니에 담긴 상태: 1, 장바구니에서 제외된 상태: 0), [상품_ID](외래키), [회원_ID](외래키));
원하는 상품 장바구니에서 제거	DELETE FROM cart WHERE product_id=[유저입력_장바구니IDX];
내 장바구니 상품 목록 보기	SELECT product.product_id, product.product_name FROM product JOIN cart ON product.product_id = cart.product_id WHERE cart.user_id=[접속한 유저의 IDX]

4.2.3 찜하기 기능

테이블	**SQL** 쿼리문
찜하기 추가	INSERT INTO `like`(like_idx, creation_date, modification_date, status, product_id, user_id) VALUES([자동생성](기본키),[생성 일자](now() 내장 함수 사용),[수정 일자](now() 내장 함수 사용), [찜하기 상태](활성화: 1, 비활성화: 0), [상품IDX](외래키), [회원IDX](외래키));
찜하기 제거	DELETE FROM lonua.like WHERE like_idx =[유저입력_찜하기IDX];
찜하기 목록 보기	SELECT * FROM lonua.like WHERE like_idx =[접속한 유저의 IDX];

4.2.4 온도**,** 카테고리**,** 스타일 별 검색 기능

테이블	**SQL** 쿼리문
온도로 상품 검색	select product.product_idx FROM product JOIN proper_temperature ON product.proper_temperature=proper_temperature.temperature WHERE proper_temperature.temperature=[유저입력 온도];
카테고리 상품 검색	select product.product_idx, product.product_name FROM product JOIN category ON product.product_idx=category.category_idx WHERE category.category_idx=[유저입력_카테고리];
스타일 별 상품검색	select product.product_idx FROM product JOIN style ON product.style1=style.style_idx WHERE style.style_idx=[유저입력 스타일];

4.2.5 리뷰 기능

테이블	**SQL** 쿼리문
리뷰 입력	INSERT INTO review(review_idx, review_content, review_photo, creation_date, modification_date) VALUE([ID자동 생성],[유저입력_리뷰내용](문자열),[유저입력_착장사진](사진을 저장한 경로), [장바구니 추가 시간](now() 내장 함수 사용),[장바구니 변경 시간](now() 내장 함수 사용));
상품에 대한 리뷰 조회	select `user`.user_name, review.review_content FROM review JOIN `order` ON review.order_id = `order`.order_idx JOIN `user` ON `user_idx` = `order`.user_id WHERE `order`.product_id=[조회 중인 상품IDX];
자신이 작성한 리뷰 수정	#자신이 작성한 리뷰 목록 조회 / 그 리뷰들에는 수정 버튼이 생긴다. select review.review_idx FROM review JOIN `order` ON review.order_id = `order`.order_idx JOIN `user` ON `user_idx` = `order`.user_id WHERE `order`.product_id=[조회 중인 상품IDX] AND `order`.user_idx = [접속한 유저의 IDX]; #클릭한 리뷰의 내용을 수정 update review set review_content=[유저입력_리뷰내용](문자열) where review_idx=[수정하려는 리뷰의 IDX];
리뷰 삭제	delete from review where review_idx=[유저입력_리뷰IDX];

4.2.6 문의 기능

테이블	**SQL** 쿼리문
문의 하기	INSERT INTO question(user_id, product_id, question_type, question_content, answer_status, secret_status) VALUE([유저IDX], [상품IDX], [유저입력_문의타입](문자열), [유저입력_문의내용](문자열), [문의 상태] (활성화 :'1', 비활성화:'0'), [유저입력_비밀글 여부](활성화 : '1', 비활성화 '0'));
내 문의 목록 조회	SELECT question_idx, question_type, question_content FROM question WHERE question_idx = 0;
문의 수정	UPDATE question SET question_content = [바꿀 문의 내용] WHERE question_idx = 1;
문의 삭제	DELETE FROM question WHERE question_idx = 1;

4.2.7 쿠폰 사용 기능

테이블	**SQL** 쿼리문
유저가 실행한 주문 정보	select `order`.order_idx, `user`.user_id, `product`.product_idx, `product`.price From `order` join `user` on `user`.user_idx = `order`.user_id join `product` on `product`.product_idx = `order`.product_id where `order_idx` = [유저입력_주문결제]
유저가 가진 쿠폰 출력	select coupon.coupon_idx , coupon.coupon_name, coupon.coupon_discount_rate, coupon.usedeadline from coupon join user on user.user_idx = coupon.user_id where user.user_idx = [회원 식별 번호];
유저가 선택한 쿠폰의 정보	select coupon.coupon_idx , coupon.coupon_name, coupon.coupon_discount_rate, coupon.usedeadline from coupon where coupon.coupon_idx = [유저입력_쿠폰ID];
결제가 완료된 후	update `lonua`.`coupon` SET coupon.status = 0 where coupon.coupon_idx = [유저입력_쿠폰ID];

4.2.8 등급 할인 사용 기능

테이블	**SQL** 쿼리문
주문 정보에서 가격 정보와 할인율 가져오기	select `order`.order_idx, `user`.user_id, `product`.product_idx, `product`.price, `user_class`.class_type,`user_class`.discount_rate From `order` join `user` on `user`.user_idx = `order`.user_id join `user_class` on `user`.user_class = `user_class`.class_idx join `product` on `product`.product_idx = `order`.product_id where `order`.order_idx = [할인 받으려는 주문IDX];

4.2.9 판매자

테이블	**SQL** 쿼리문
판매자(Brand) 추가	INSERT INTO brand(brand_id, brand_pw, brand_photo, brand_introduction, brand_style, phone_number, business_registration, mail_order_business_report_card, bank_account_number, return_address,return_cost, return_courier, business_address, creation_date, modification_date, status) VALUE([유저 입력_브랜드 ID](판매자 전용 접속을 위한 ID), [유저 입력_브랜드 PW](판매자 전용 접속을 위한 PW), [브랜드 사진](사진이 위치한 경로), [유저 입력_브랜드 소개](문자열), [유저 입력_브랜드_스타일](문자열) ,[유저 입력_전화번호], [유저 입력_사업자등록증 사진](사진이 위치한 경로),[유저 입력_통신판매등록증_사진](사진이 위치한 경로), [유저입력_계좌번호](문자열), [유저입력_환불주소](문자열), [유저입력_환불비용], [유저입력_환불방법],[유저입력_사업장주소](문자열), [등록 일시](now() 내장함수 이용),[정보 수정 일시](now() 내장함수 이용), [계정 상태]);
판매자(Brand) 정보 수정	UPDATE brand SET [속성]=[유저입력_속성값]' WHERE brand_idx=[접속한 판매자의 IDX];
판매자(Brand) 정보 삭제	DELETE FROM brand WHERE brand_idx=[유저입력_삭제될 판매자의 IDX];

4.2.10 판매상품

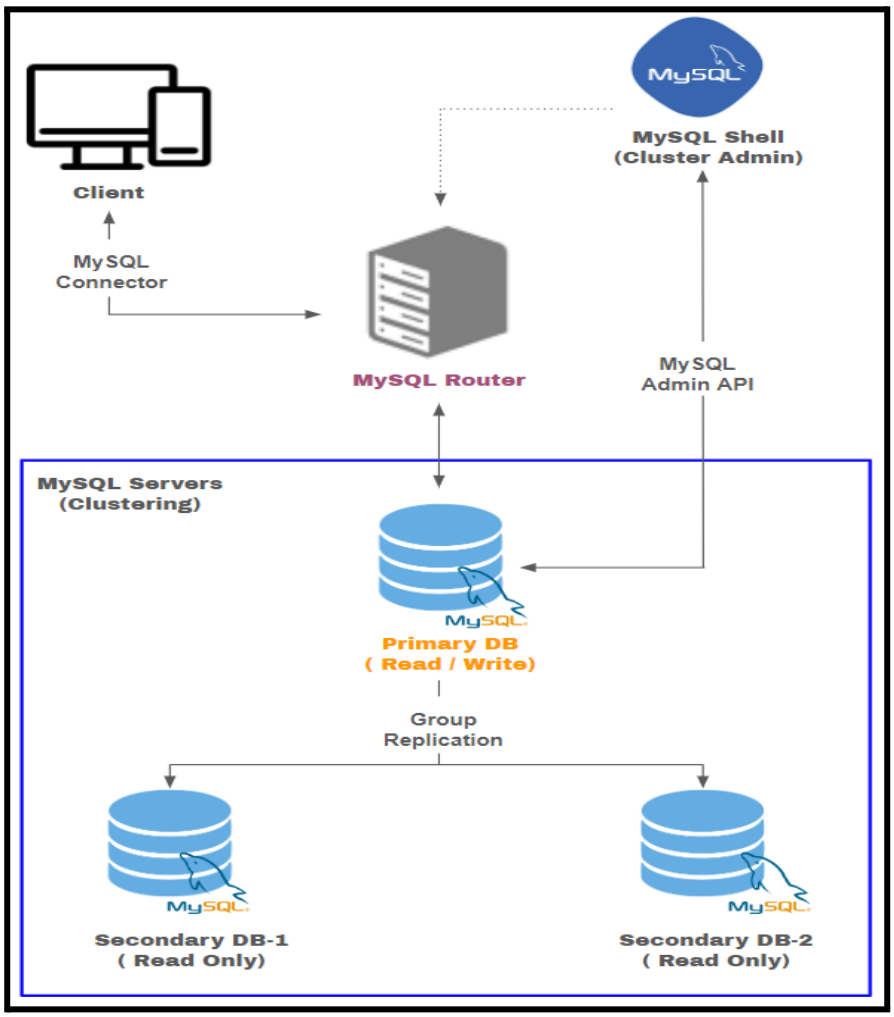
테이블	**SQL** 쿼리문
판매상품 등록	INSERT INTO product (brand_id,product_name,quantity,price,product_option,shoulder_width,chest_size,arm_length,top_length,waistline,hip_circumference,thigh_circumference,crotch_length,hem_length,total_bottom_length,category_idx,creation_date,modification_date, status,category_id,proper_temperature1,proper_temperature2,proper_temperature3,style1,style2,style3) values([상품등록자IDX], [유저입력_상품이름](문자열), [유저입력_수량](INT), [유저입력_가격](INT), [유저입력_옵션](문자열), [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_상품치수], [유저입력_카테고리](외래키),[등록 일시](now() 내장함수 이용),[정보 수정 일시](now() 내장함수 이용), [상품 상태](활성화: 1 , 비활성화 0),[유저입력_카테고리](외래키), [유저입력_적정온도1](외래키), [유저입력_적정온도2](외래키),[유저입력_적정온도3](외래키),[유저입력_스타일1](외래키),[유저입력_스타일2](외래키),[유저입력_스타일3](외래키));
모든 판매상품 목록보기	SELECT product.product_idx, product.product_name , brand.brand_name FROM product JOIN brand ON product.brand_id = brand.brand_id
판매상품 수정	UPDATE product SET [속성_이름]=[유저입력_속성값] WHERE product_idx=[유저입력_수정할 상품];
판매상품 삭제	DELETE FROM product WHERE product_idx=[상품IDX];

4.2.11 매장

테이블	**SQL** 쿼리문
자신의 매장 목록보기	SELECT store_address FROM store;
자신의 매장 위치 입력	INSERT INTO store(store_idx, store_address, brand_idx) VALUES([자동입력](INT), [매장주소](문자열), [매장 추가한 브랜드IDX]);
자신의 매장 정보수정	UPDATE store SET store_address =[바뀐 매장 정보] WHERE store_idx = [바꿀 매장의 id];
매장 삭제	DELETE FROM store WHERE store_address = [매장의 주소];

4.3 시스템 아키텍처

4.3.1 구성도



< 서버 구성 >

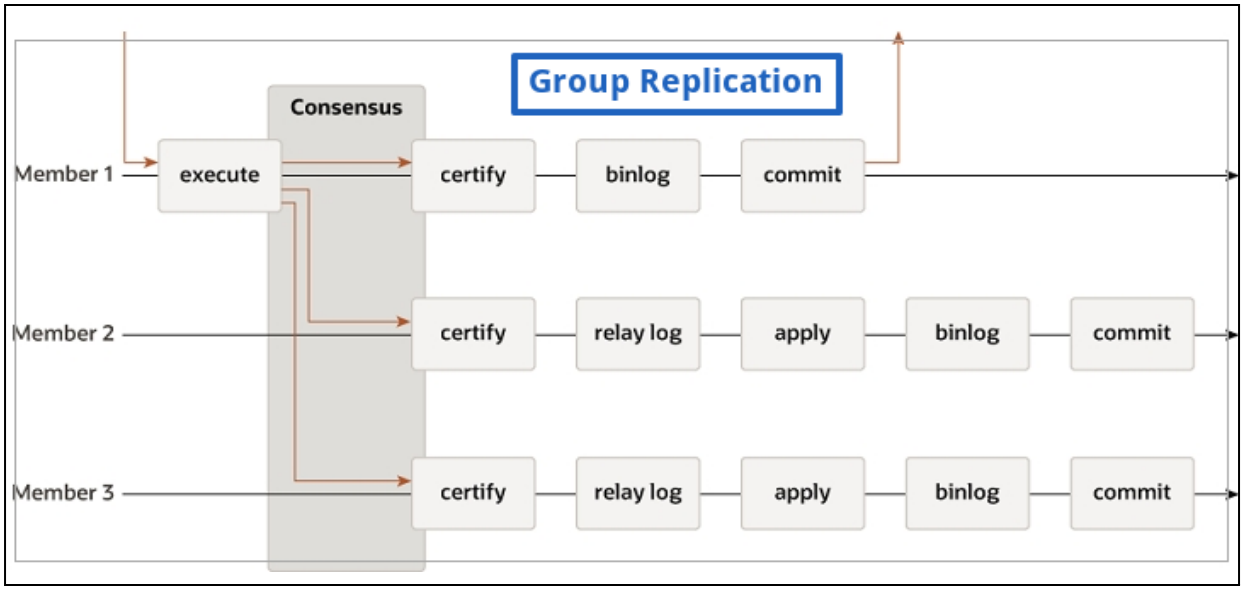
PC명	설 명	**PC** 사양
Router	라우터	CPU : 1core Memory : 1GB HardDisk : 20GB
Primary DB	기본 Primary DB	
DB1	Secondary DB-1	
DB2	Secondary DB-2	

4.3.2 DB Clustering 설정 사유

"LONUA" 는 온라인 쇼핑몰의 특성 상 판매자가 상품을 등록(쓰기 작업) 하는 횟수에 비해, 구매자가 **상품을 검색(읽기 작업)하는 횟수가 더욱 많이 발생**한다.
또한, 쇼핑몰에서 다양한 할인 행사를 진행할 때 **많은 사용자가 동시에 웹 사이트로 접속**할 시 트래픽 과부하가 발생할 수 있으므로, 이용자의 안정적인 쇼핑 활동 보장 및 주문한 상품들에 대한 데이터의 신뢰성이 높아야 하기때문에, 서버를 구성할때 비동기 방식으로 운영되어 일관성 있는 데이터의 보장이 어려운 "Replication" 방식이 아닌 "Clustering" 방식으로 구성하였다. (총 4개의 서버로 1개는 라우터, 3개는 Group Replication을 이용하여 1개의 Primary 서버와 2개의 Secondary 서버로 구성 [Active-Standby])

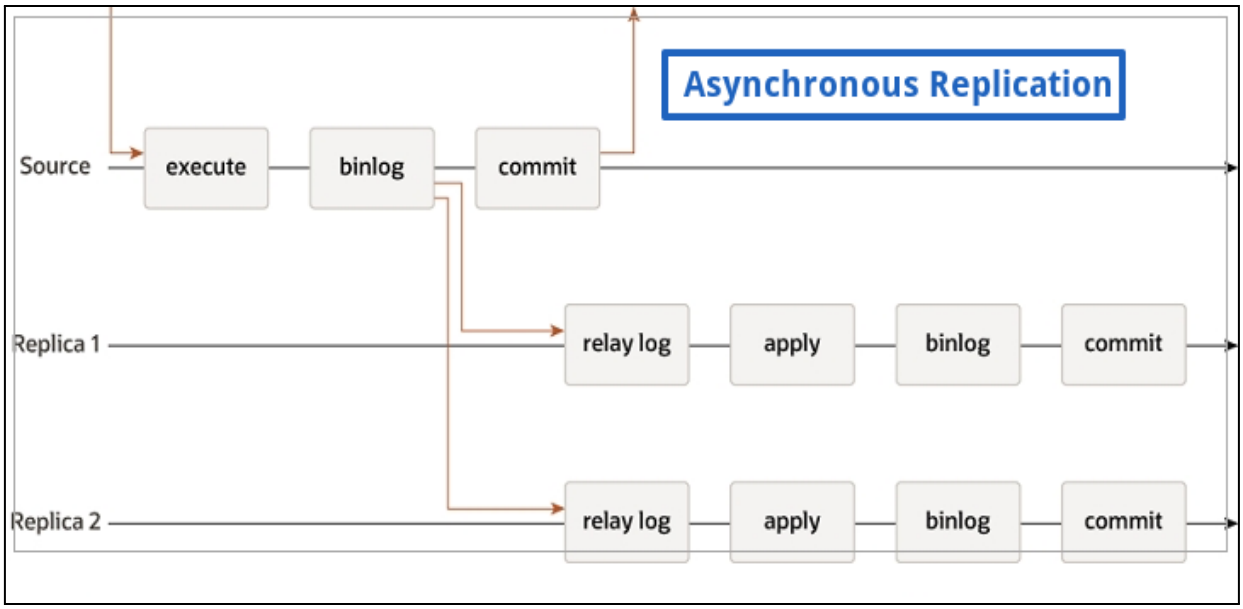
[참고 자료]

> Clustering 방식 (Group Replication / 동기 방식)



=> Router 에서 데이터 변경 작업 발생 시 Primary 서버와 Secondary 서버 2대 모두에서 동시에 실행 명령이 입력되고 같은 작업이 이루어지게 되므로 **데이터의 신뢰성이 높다.**

> Replication 방식 **(Master/Slave /** 비동기 방식**)**



1. Master 서버에서 데이터 변경사항 발생 시 “binlog” 에 기록이 되고, Commit 실시 후 트랜잭션을 종료한다.
2. “binlog” 에 기록이 된 데이터를 참조해서 Slave 서버가 “relaylog” 가 기록을 하고 기록된 “relaylog”를 통해 Slave 서버가 이벤트를 실행한다.
3. 이벤트 종료 후 Slave 서버의 “binlog” 에 데이터를 저장하고 Commit 실시 후 트랜잭션을 종료한다.

=> 따라서, Master 서버와 Slave 서버의 트랜잭션이 **서로 동기화 되어 있지 않기 때문에** Master 서버에 문제가 발생하면 Slave 서버로 전환되는 과정, Slave 서버에 문제가 발생할 시 Master 서버의 이벤트를 가져올 때 **누락된 데이터가 있을 수 있다.**